Ed Marczak tells us: "What's in my sed"

Net Monitoring with Nagios How To Easily Set It Up

MacTech Magazine December • 2005

MACTECH®

The Journal of Macintosh Technology

MacTech 2005 High-Tech Gift Guide

Gift buyers advice...the MacTech way!

"Why Am I The Only Woman At This Table?" In The Trenches with Schoun Regan

Deconstructing RSS 2.0 Understanding How Feeds Work

State Property II Opening QT Files with Properties

Building Table Views with AppleScript Studio

$8.95 US,
$12.95 Canada
ISSN 1067-8360
Printed in U.S.A.

12
0 32128 74887 8


KNOW WHO'S WATCHING YOU.
PROTECT YOUR PRIVACY
MONITOR COMMUNICATIONS
SECURELY ERASE FILES
BLOCK ANNOYING WEB ADS
SUPPORT FOR APPLE MAIL
Allume Systems
KNOW WHO'S WATCHING YOU
STOP SPYWARE
PRIVACY PROTECTION security from prying eyes
MONITOR COMMUNICATIONS to and from your Mac
BLOCK ANNOYING ADS and pop-up windows
PROTECT YOUR PRIVACY
INTERNET CLEANUP 3.0
INTERNET CLEANUP
PROTECT YOUR PRIVACY
Allume Systems
A Division of Smith Micro Software
Available from your favorite retailers and catalogs. Download a FREE Trial at Allume.com!
One hundred percent 10.4 "Tiger" compatible! Download a FREE trial version of Internet Cleanup at http://www.allume.com/mac/cleanup/ today!
®, TM, and © 2005 Allume Systems, Inc. Mac OS X 10.3 "Panther" is required for Internet Cleanup 3.0.

TABLE OF CONTENTS

MACTECH 2005 HIGH-TECH GIFT GUIDE

By Michael R. Harvey, Lesa Snider, and Aaron Adams

MacTech Magazine (ISSN: 1067-8360 / USPS: 010-227) is published monthly by Xplain Corporation, 850-P Hampshire Road, Westlake Village, CA 91361-2800. Voice: 805/494-9797, FAX: 805/494-9798. Domestic subscription rates are $47.00 per year. Canadian subscriptions are $59.00 per year. All other international subscriptions are $97.00 per year. Domestic source code disk subscriptions are $77 per year. All international disk subscriptions are $97.00 a year. Please remit in U.S. funds only. Periodical postage is paid at Thousand Oaks, CA and at additional mailing office.

POSTMASTER: Send address changes to **MacTech Magazine**, P.O. Box 5200, Westlake Village, CA 91359-5200.

Stay In Control Wherever You Go.

Timbuktu® Pro

Whether you're at home or at work, using a Mac OS or Windows platform, Timbuktu Pro allows you to operate distant computers as if you were sitting in front of them. Transfer files and folders quickly and easily, and communicate by instant message, text chat, or voice intercom.

eCare

eCare's web-based remote access capabilities enable any user to securely place their virtual eyes and fingers directly on the remote client's desktop for collaboration, problem resolution, and improved customer satisfaction. Using only a web browser, eCare's secure and fast screen sharing, file transfer, live chat, URL push and live session recording make it the ideal solution for the conference room, classroom, or help desk – live, on demand.

Download an evaluation or buy it online: www.netopia.com
Call us at 1-800-485-5741

timbuktu® pro · eCare

netopia.

MACTECH

Communicate With Us

Department E-Mails

Orders, Circulation, & Customer Service
cust_service@mactech.com

Press Releases
press_releases@mactech.com

Ad Sales
adsales@mactech.com

Editorial
editorial@mactech.com

Online Support
online@mactech.com

Accounting
accounting@mactech.com

Marketing
marketing@mactech.com

General
info@mactech.com

Web Site
http://www.mactech.com

In this electronic age, the art of communication has become both easier and more complicated. Is it any surprise that we prefer **e-mail?**

If you have any questions, feel free to call us at 805/494-9797 or fax us at 805/494-9798.

If you would like a subscription or need customer service, feel free to contact MacTech Magazine Customer Service at 877-MACTECH

We love to hear from you! Please feel free to contact us with any suggestions or questions at any time.

Write to letters@mactech.com or editorial@mactech.com as appropriate.

ARTICLES / DEPARTMENTS

"Why Am I The Only Woman At This Table?"

For this month's installment of Schoun Regan's new monthly column, "In The Trenches", Schoun took some time out of his busy schedule at this year's WWDC to have a chat with Andrina Kelly, a Senior IT staff member at C.O.R.E Feature Animation in Toronto. We're sure you'll enjoy their discussion!

Schoun Regan: How long have you been working with Macs?

Andrina Kelly: Since my first 512k when I was very young.

SR: Does your networking background comes from those early years too?

AK: No, I was working in the UK at a classical CD production company full of Macs and someone had to take on the responsibility and that was me.

SR: Do you think a lot of people start that way?

AK: Probably. I think there are a lot of self taught learners who came on the same way.

SR: Did you pick up your programming at the same time?

AK: Probably over the past two years. It's something I always avoided because it was something my mother did it. It's something I should have taken more seriously when I was younger.

SR: Do you allow scripts to interoperate between each other?

AK: Our rendering department needed a pop up messaging system so they could alert the floor when there were issues or they needed them to know something quickly. All of our rendering pods use LINUX, so the best way for them was to send out a shell script but that also incorporated into an AppleScript on the user machines so it would pop up a message. I don't know of any commercial product that would have covered it, because this covers both LINUX and Mac OS X.

SR: So CORE benefits from this. Tell me more about CORE.

AK: CORE is quite a diverse company. Recently they worked with a San Francisco company and put together a proposal for Disney to put out a feature animation film. We've been working on it for about 2 _ -3 years now and it is just coming up to completion and should be released in April 2006 called The Wild.

SR: Your role at CORE is?

AK: I'm one of the System Administrators with main focus on the Macs.

SR: What's your biggest day-to-day struggle? Your biggest headache?

AK: Interoperability. Allowing someone who's used to working on LINUX, used to working in a terminal environment, going over to a Mac, opening up the Terminal and going, "I'm going to go into this directory and open up this file" and having it work exactly the same for them.

SR: That said, how easy did the LINUX users take to the Mac?

AK: From what I've heard, they LOVE it! (Laughs) I can name several people that I've been working with in the last couple of years who've gone out and got a Mac.

SR: If you could take one thing from the LINUX environment and move it over to Mac OS X to convert those die hards, what would that one thing be?

AK: (Silence.....laughter and a long pause) I think the X windows to Finder transition is a little awkward for some people.

SR: It is easy being female in a male dominated industry?

AK: I was just talking about this with one of my colleagues today. It's a question I love to ask whenever I'm in a large environment of men, "Why am I the only woman at this table?" I find that women are a lot more patient and a lot more programmatic about

things. I know for troubleshooting especially the number of women I've seen troubleshooting tend get to the answer a lot quicker than men. I find men get frustrated more quickly.

SR: Any advice?

AK: Don't get discouraged and don't be put off.

SR: Your OS X Servers, specifically render farms?

AK: Web Services, LDAP Masters and Replicas, My SQL, PHP, and failover setup All authentication is done with the OS X servers.

SR: Using Kerberos?

AK: Not at the moment. The infrastructure is all there though.

SR: How do the Xserve RAIDs perform?

AK: I have not had a failed drive in over two years. I couldn't be more pleased with the hardware from Apple. We had the first G5 cluster nodes in Canada. Since we've turned them on, I have not had a hardware failure.

SR: Being in the center of the IT industry in Canada in Toronto, what trends do you see?

AK: Anything media based is going to be strongly Mac. I've recently seen a few places like lawyers offices and psychiatry offices using Macs, which is nice to see and it shows a lot of faith that there's going to be a lot more progression towards the Mac. And in these smaller offices they don't have a full time tech staff and the fact that they can manage their own systems and their own server I bodes well for Apple I think.

SR: I have a couple sentences that I want you to finish, ready?

AK; Ready.

SR: I wish Mac OS X Server had...

AK: A better interaction between the configuration files and the GUI. I'd really love to see it's configuration file and what the GUI is changing and where the file is located.

SR: So almost if you had Server Admin open that had a drawer, would show you the configuration file and path, and highlight the section of the configuration file when you would make adjustments in the GUI. This of course for most services to start.

AK: Right.

SR: I wish Mac OS X had...

AK: Hmmmm. I find that moving between networks and environments is not as seamless as I'd like it to be. A better awareness of its location. Something like Location Manager. NFS performance is something else that I am crying out for. The performance from NFS is- I'd love to see it so much better than it is. A straight copy is so significantly slower on OS X than on other platforms.

AK: Last question. And you will be starting our database here as I this is a question that I will ask all our guests in the future, what are your top five favorite movies? Movies that you could watch over and over and over again. No order is necessary, just the top five.

AK: (Laughs) ok, let me think....(long pause)...they would be: The Red Violin, The English Patient, Sideways, The Italian Job (1969 version), The Wild (Released April 2006!).

SR: Andrina, thanks very much for speaking with me today.

AK: Thank you.

MT

Vital Stats

Years in IT industry : 5

Place of Employment: C.O.R.E. Feature Animation/C.O.R.E. Digital Pictures 450-500 people. Toronto, Canada

Computers: Approximately 1 per user. 1/3 Macs 2/3 LINUX 500 Blade Dual Processor IBM Render Farm 32 Dual 2G Xserve Cluster nodes for Shake rendering and Final Cut Pro rendering BlueArc, NetApp appliances, and Xserve RAIDs totaling about 100TB.

Main Applications: Houdini, Final Cut Pro, Shake

Programming Languages: Shell Scripting, AppleScript, PYTHON

About The Author

Schoun P. Regan is CEO of ITInstruction.com, which specializes in Mac OS X training and consulting. He speaks regularly to CEOs and CFOs on how to control IT department spending, the myths surrounding cross-platform integration, and the lunacy of expected lost revenue stemming from a culture bred to tolerate IT staff and operating system inadequacies as "normal". He seeks to change self-fulfilling IT departments that breed complacency for their jobs and contempt for the end user, neither of which are conducive to business.

STATE PROPERTY 2

OPENING QUICKTIME MOVIES USING PROPERTIES

In the previous QuickTime Toolkit article ("State Property" in MacTech, November 2005), we learned how to work with the QuickTime properties functions introduced in QuickTime 6.4 and considerably expanded in QuickTime 7. We saw how to get and set movie properties using those functions, and we saw how to install a property listener so that our applications can be informed about changes to some of those movie properties. Currently, only two small sets of movie properties are gettable or settable or listenable using those functions. These are a handful of visual properties (hue, saturation, brightness, and contrast) and a smaller handful of audio properties (gain, mute, and balance). So why is the header file Movies.h chock full of identifiers for other property classes and other property types within those classes?

The answer is that those additional properties are intended for use by other functions, and in particular by the `NewMovieFromProperties` function, which was introduced in QuickTime 7. `NewMovieFromProperties` is intended as a replacement for the plethora of existing "`NewMovieFrom`" functions, including

`NewMovie`, `NewMovieFromFile`, `NewMovieFromHandle`, `NewMovieFromDataRef`, `NewMovieFromStorageOffset`, `NewMovieFromDataFork`, `NewMovieFromDataFork64`, `NewMovieFromScrap`, and `NewMovieFromUserProc`.

The basic idea is that we first specify an array of properties that we'd like our new movie to have, and then we call `NewMovieFromProperties`. This allows us to open a movie that has exactly the properties we desire, without having to rely on QuickTime to establish some default set of properties that we later need to override. It also allows us to specify properties that cannot be specified using the existing functions. For instance, the ***constant pitch audio*** setting (whereby the audio pitch remains constant even when the playback rate of a movie increases or decreases, thus avoiding the infamous "chipmunk" effect when fast-forwarding through a movie) must be set at the time a movie is created, and `NewMovieFromProperties` is the only movie-opening function that allows us to specify a setting for that property.

In this article, we'll see how to work with `NewMovieFromProperties`. We'll take a look at the classes of properties that we can pass to it and see how to

set movie properties not otherwise settable using the existing `NewMovieFrom` functions. This topic might seem vaguely familiar to you, as we touched on a very similar programming model when investigating the `initWithAttributes:error:` method in the `QTMovie` class in QTKit (see "Back to the Future, Part III" in ***MacTech***, July 2005). In fact, `initWithAttributes:error:` internally calls `NewMovieFromProperties`, as you might easily have guessed.

Input Properties

Let's begin by taking a look at the declaration of `NewMovieFromProperties`. In the `Movies.h` file shipped with QuickTime 7, we see essentially this:

```
OSStatus NewMovieFromProperties (
   ItemCount                     inputPropertyCount,
   QTNewMoviePropertyElement*  inputProperties,
   ItemCount                     outputPropertyCount,
   QTNewMoviePropertyElement*  outputProperties,
   Movie *                       theMovie);
```

As you can see, this function takes as input an array of `QTNewMoviePropertyElement` structures (`inputProperties`) and the number of elements in that array (`inputPropertyCount`). These properties describe how to instantiate the movie. If successful, this function returns a QuickTime movie identifier in the location pointed to by the `theMovie` parameter. It may also return to the caller a different array of properties (`outputProperties`), which provide additional information about the newly-created movie. These output properties, for instance, may indicate whether a data reference passed in the `inputProperties` array was changed during the process of opening the movie. (More on output properties later.)

The `QTNewMoviePropertyElement` structure is defined like this:

```
struct QTNewMoviePropertyElement {
   QTPropertyClass          propClass;
   QTPropertyID             propID;
   ByteCount                propValueSize;
   QTPropertyValuePtr       propValueAddress;
   OSStatus                 propStatus;
};
```

The first two fields of this structure are the class and the ID of a movie property; if you have read the previous two ***QuickTime Toolkit*** articles, these items should be clear enough. The third and fourth fields indicate the size and location of the value of that property.

For input properties, the final field of the QTNewMoviePropertyElement structure, propStatus, is set by NewMovieFromProperties to a status value that indicates whether the specified property was successfully set on the new movie. In general, the value of this field will be set to 0 (noErr). But occasionally a non-zero result will be returned in that field. For example, if you mistakenly pass in a data value that is not the size that QuickTime is expecting for the class and ID you specify, the value –2184 (kQTPropertyBadValueSizeErr) may be returned in that field. Similarly, if you specify a property that cannot be set, then the value –2191 (kQTPropertyReadOnlyErr) will be returned.

The simplest possible way to call NewMovieFromProperties is to pass in no input properties and to request no output properties, like this:

```
err = NewMovieFromProperties(0, NULL, 0, NULL,
&movie);
```

This is effectively the same as calling NewMovie with its flags parameter set to 0 — not particularly useful, but sometimes necessary if you just want to create a new empty movie with all the default characteristics. We're more likely to call NewMovieFromProperties passing in a properties array that at least includes a movie location and some additional properties. Let's see how to do that.

Specifying a Movie Location

The location of a movie's data is specified by adding to the input properties array an element with the kQTPropertyClass_DataLocation class. Currently these property IDs are supported:

```
enum {
kQTDataLocationPropertyID_DataReference  = 'dref',
  kQTDataLocationPropertyID_CFStringNativePath
= 'cfnp',
  kQTDataLocationPropertyID_CFStringPosixPath
= 'cfpp',
  kQTDataLocationPropertyID_CFStringHFSPath
='cfhp',
  kQTDataLocationPropertyID_CFStringWindowsPath
= 'cfwp',
  kQTDataLocationPropertyID_CFURL            = 'cfur',
  kQTDataLocationPropertyID_QTDataHandler
='qtdh',
  kQTDataLocationPropertyID_Scrap            = 'scrp',

kQTDataLocationPropertyID_LegacyMovieResourceHandle
= 'rezh',
  kQTDataLocationPropertyID_MovieUserProc  = 'uspr',
  kQTDataLocationPropertyID_ResourceFork = 'rfrk',
  kQTDataLocationPropertyID_DataFork      = 'dfrk'
};
```

For example — starting with an easy case — we can open a new movie that uses data on the scrapbook (or Cocoa pasteboard) by using the kQTDataLocationPropertyID_Scrap ID, as shown in Listing 1.

Listing 1: Loading a movie from the scrapbook/pasteboard

```
QTNewMoviePropertyElement props[1] = {{0}};
Movie movie = NULL;

props[0].propClass = kQTPropertyClass_DataLocation;
props[0].propID = kQTDataLocationPropertyID_Scrap;
props[0].propValueSize = 0;
props[0].propValueAddress = NULL;

err = NewMovieFromProperties(1, props, 0, NULL,
&movie);
```

Notice that we do not need to assign any non-zero value to the propValueAddress field, since the property ID uniquely identifies the location of the movie data. Listing 1 provides a reasonable approximation of the existing NewMovieFromScrap function.

Listing 2 shows a slightly more interesting example, which opens a movie specified by a URL, in this case a CFURL.

Listing 2: Loading a movie from a URL

```
QTNewMoviePropertyElement props[1] = {{0}};
Movie movie = NULL;

props[0].propClass = kQTPropertyClass_DataLocation;
props[0].propID = kQTDataLocationPropertyID_CFURL;
props[0].propValueSize = sizeof(CFURLRef);
props[0].propValueAddress = &cfurl;

err = NewMovieFromProperties(1, props, 0, NULL,
&movie);
```

And Listing 3 shows the most general case, where the movie data location is specified by a data reference. In this case, we need to pass the address of a DataReferenceRecord, declared like this:

```
struct DataReferenceRecord {
  OSType              dataRefType;
  Handle              dataRef;
};
```

Listing 3: Loading a movie from a URL data reference

```
QTNewMoviePropertyElement props[1] = {{0}};
DataReferenceRecord dRefRec;
Movie movie = NULL;

dRefRec.dataRefType = URLDataHandlerSubType;
dRefRec.dataRef = url;

props[0].propClass = kQTPropertyClass_DataLocation;
props[0].propID =
kQTDataLocationPropertyID_DataReference;
props[0].propValueSize = sizeof(dRefRec);
props[0].propValueAddress = &dRefRec;

err = NewMovieFromProperties(1, props, 0, NULL,
&movie);
```

Specifying Movie Properties

So far, this should all be straightforward: for any particular movie data locator ID, we just need to set the `propValueSize` and `propValueAddress` fields appropriately. There should be at most one data locator property in the array we pass to `NewMovieFromProperties`. But there can also be other kinds of properties, including ***movie instantiation properties*** (whose class is `kQTPropertyClass_MovieInstantiation`) and ***new movie properties*** (whose class is `kQTPropertyClass_NewMovieProperty`). Here are the currently-defined movie instantiation input properties, which govern how QuickTime instantiates a movie:

```
enum {

kQTMovieInstantiationPropertyID_DontResolveDataRefs
= 'rdrn',

kQTMovieInstantiationPropertyID_DontAskUnresolvedDat
aRefs = 'aurn',

kQTMovieInstantiationPropertyID_DontAutoAlternates
= 'aaln',

kQTMovieInstantiationPropertyID_DontUpdateForeBackPo
inters = 'fbpn',
  kQTMovieInstantiationPropertyID_AsyncOK           =
'asok',
  kQTMovieInstantiationPropertyID_IdleImportOK =
'imok',
kQTMovieInstantiationPropertyID_DontAutoUpdateClock
= 'aucl'
};
```

And here are the currently defined new movie properties, which provide additional settings for a new movie:

```
enum {

  kQTNewMoviePropertyID_DefaultDataRef          = 'ddrf',
  kQTNewMoviePropertyID_Active                  = 'actv',
  kQTNewMoviePropertyID_DontInteractWithUser    =
'intn'
};
```

These two sets of properties mirror the `newMovie` flags specifiable as a parameter to the `NewMovie` and similar functions:

```
enum {
  newMovieActive                        = 1 << 0,
  newMovieDontResolveDataRefs           = 1 << 1,
  newMovieDontAskUnresolvedDataRefs     = 1 << 2,
  newMovieDontAutoAlternates            = 1 << 3,
  newMovieDontUpdateForeBackPointers    = 1 << 4,
  newMovieDontAutoUpdateClock           = 1 << 5,
  newMovieAsyncOK                       = 1 << 8,
  newMovieIdleImportOK                  = 1 << 10,
  newMovieDontInteractWithUser          = 1 << 11
};
```

For example, to open a movie specified by a URL so that the movie data loads asynchronously and so

that the resulting movie is active, we could execute the code in Listing 4.

Listing 4: Loading a movie from a URL with additional properties

```
QTNewMoviePropertyElement props[3] = {{0}};
DataReferenceRecord dRefRec;
Movie movie = NULL;
Boolean isActive = true;
Boolean isAsync = true;
long num = 0;

dRefRec.dataRefType = URLDataHandlerSubType;
dRefRec.dataRef = url;

props[0].propClass = kQTPropertyClass_DataLocation;
props[0].propID =
kQTDataLocationPropertyID_DataReference;
props[0].propValueSize = sizeof(dRefRec);
props[0].propValueAddress = &dRefRec;
num++;

props[1].propClass =
kQTPropertyClass_MovieInstantiation;
props[1].propID =
kQTMovieInstantiationPropertyID_AsyncOK;
props[1].propValueSize = sizeof(isAsync);
props[1].propValueAddress = &isAsync;
num++;

props[2].propClass =
kQTPropertyClass_NewMovieProperty;
props[2].propID = kQTNewMoviePropertyID_Active;
props[2].propValueSize = sizeof(isActive);
props[2].propValueAddress = &isActive;
num++;

err = NewMovieFromProperties(num, props, 0, NULL,
&movie);
```

Output Properties

As noted earlier, `NewMovieFromProperties` can also return a set of properties to the caller, which indicate additional information about the newly-opened movie. Currently there are two such output properties:

```
kQTMovieResourceLocatorPropertyID_LegacyResID
kQTMovieResourceLocatorPropertyID_LegacyResName
```

These indicate the resource ID of the movie and the name of the movie resource. The resource name is generally not terribly useful, but the resource ID can be useful in determining whether the movie atom was loaded from the file's data fork (returned value is –1) or the resource fork (returned value is greater than 0), or whether there was no movie atom in the storage container (returned value is 0).

We can obtain an output property by passing in a second array of `QTNewMoviePropertyElement` structures, as shown in Listing 5.

Listing 5: Getting an output property

```
QTNewMoviePropertyElement props[3] = {{0}};
QTNewMoviePropertyElement outProps[1] = {{0}};
Movie movie = NULL;
short resID = 0;
long num = 0;

// set-up of input properties omitted

outProps[0].propClass =
        kQTPropertyClass_MovieResourceLocator;
outProps[0].propID =

kQTMovieResourceLocatorPropertyID_LegacyResID;
outProps[0].propValueSize = sizeof(resID);
outProps[0].propValueAddress = &resID;

err = NewMovieFromProperties(num, props, 1,
outProps,
        &movie);
```

On successful completion of this code, the local variable `resID` will contain the resource ID of the movie resource.

Conclusion

In this article, we've learned how to use the `NewMovieFromProperties` function introduced in QuickTime 7 as a replacement for the array of existing `NewMovieFrom` functions. We've seen how to specify the location of the movie data and how to set default properties on the new movie. We've also seen how to get values of certain properties back from `NewMovieFromProperties`.

In the next several articles, we'll continue investigating `NewMovieFromProperties`. In particular, we'll take a look at the properties associated with the `kQTPropertyClass_Context` property class, which allows us to set ***media context properties*** of a movie. We use these properties to create movies that render into a visual context (such as an OpenGL texture buffer) or to a particular audio device.

MT

About The Author

Tim Monroe is a member of the QuickTime engineering team at Apple. You can contact him at monroe@mactech.com. The views expressed here are not necessarily shared by his employer.

Deconstructing RSS 2.0

Understanding How RSS Feeds Work

By Dave Woolridge

If you're one of the millions of people who maintain your own weblog, then odds are, you're probably already familiar with RSS or Atom feeds since most blog tools include support for offering your blog as a syndicated feed. If you don't write a blog, then you've undoubtedly seen the RSS icon displayed online (see **Figure 1**) or have been invited by web sites to subscribe to their free RSS feeds. In fact, it's such a hot technology these days that you would have had to have been living "off the Grid" for the last few years to have not heard about RSS.

With aggressive spam filters making e-mail communication difficult for even legitimate marketers and businesses, web feeds have become a safe and dependable method for you to successfully deliver news to your audience, as well as allow third-party sites to syndicate your feed content for expanded reach to new viewers. It's a win-win situation for everyone involved. Users can subscribe to only the feeds they wish to receive and third-party sites are provided with free content for their sites that ultimately drives additional traffic to your site via your feed's links.

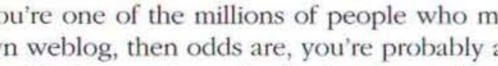


Figure 1. Typical RSS buttons seen on web sites and blogs, commonly referred to as "chicklets"

While the RSS and Atom formats are both used throughout the Internet for content syndication, this article will focus on the RSS 2.0 specification. With support for multimedia enclosures and other multi-purpose features, RSS 2.0 is quickly becoming one of the most popular flavors for content syndication, providing site owners with a powerful vehicle for delivering a lot more than just news. Have you ever subscribed to a podcast? Yup, you guessed it… podcasts are RSS 2.0 feeds.

How RSS Works

Before we dive in, let's dispel the most common misconception about web feeds. While many people use the word "broadcast" to describe online content syndication, web feeds are not transmitted like radio or television signals. They are not beamed or sent to subscribers. A web feed is nothing more than an XML document that resides on a web server. This means that your news reader software or web browser is fetching the RSS feed from a specified URL, just like it would do to access an HTML web page. The reader/browser software then parses (translates) that XML document, providing it to you in a display format that's easy to read. When using a news reader application (such as NetNewsWire or NewsFire) or an RSS-savvy browser (such as Safari or Firefox), subscribing to an RSS feed is like adding a bookmark to your favorites list. The only difference is that an RSS feed will get automatically checked for new updates on a regular basis, which requires the reader/browser to go fetch the XML document from the feed's URL at each timed interval.

XML Syntax

Since we'll be taking a look at the structure of an RSS feed and the specific functionality of each XML tag in the RSS specification, it's important to understand the basic syntax of XML (Extensible Markup Language). Like HTML code, XML consists of tags such as:

```
<title>My RSS Feed</title>
```

Unlike HTML, which helps define styles and formatting for text, XML tags strictly define the meaning and context of information, keeping all of the text neatly organized with tag names. You don't have to be a master of XML in order to write or modify your own RSS feeds, but there are a few basic rules that you should keep in mind.

- Every valid RSS feed needs to include <?xml version="1.0" encoding="UTF-8"?> as the very first tag at the top of the document. UTF-8 is what most feeds use as the text encoding and most RSS

parsers assume UTF-8 as the default if no encoding is specified, but if you need to use a different encoding for a special purpose, then change the encoding attribute accordingly.

- The nesting order of XML tags is very important in order for your code to be valid. For example, the following line is properly nested:

```
<skipDays><day>Sunday</day></skipDays>
```

  while the same line of code below will cause errors since it contains invalid nesting:

```
<skipDays><day>Sunday</skipDays></day>
```

- XML is case-sensitive, so <skipdays> is not the same as <skipDays>. While some RSS readers and browsers may be smart enough to overlook case typos, you certainly want to avoid any potential problems with your feed, so if you're modifying your RSS code by hand in a text editor, make sure your tags conform to the RSS 2.0 specification.

- Be very careful when including HTML code within XML. If you need to include HTML code within the title or description tags of a news item, you need to either enclose all of the text within a CDATA block such as:

```
    <description><![CDATA[My <b>Big</b> News]]>
</description>
```

  or convert all HTML brackets and special characters into XML-safe entities such as:

```
    <description>My &lt;b&gt;Big&lt;/b&gt;
News</description>
```

Reviewing an Example

To get a feel for what we are talking about here, let's take a look at an example RSS feed. RSS is short for "Really Simple Syndication," and as you'll see from the XML code in **Listing 1**, the format really is quite simple overall. RSS 2.0 feeds can be saved with a file extension of either .rss or .xml. **Listing 1** shows an RSS feed that includes two items. The first one is a typical news item, while the second one contains a media enclosure, similar to what you would find in a podcast.

Listing 1: An RSS 2.0 Feed Example

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0">
<channel>
  <title>SpiderWorks News</title>
  <link>http://www.spiderworks.com/</link>
  <description>Quality eBooks and Printed Books
from Respected Authors at a Great
Price!</description>
  <language>en</language>
```

```
  <copyright>Copyright 2005 SpiderWorks, LLC. All
rights reserved.</copyright>

<managingEditor>news@spiderworks.com</managingEditor
>
  <webMaster>support@spiderworks.com</webMaster>
  <rating> </rating>
  <pubDate>Mon, 24 Oct 2005 05:53:07 GMT</pubDate>
  <lastBuildDate>Mon, 17 Oct 2005 08:13:02
GMT</lastBuildDate>
  <category
domain="http://www.spiderworks.com">Books</category>
  <docs>http://blogs.law.harvard.edu/tech/rss</docs>
  <generator>SpiderWorks Staff</generator>

  <skipDays><day>Sunday</day></skipDays>
  <skipHours><hour>14</hour></skipHours>
  <ttl>30<ttl>
  <cloud domain="http://www.spiderworks.com"
port="80" path="/rpc"
registerProcedure="rssPleaseNotify" protocol="XML-
RPC" />

  <image>
    <title>SpiderWorks</title>

<url>http://www.spiderworks.com/swbadge.gif</url>
    <link>http://www.spiderworks.com/</link>
    <width>120</width>
    <height>35</height>
    <description>Quality eBooks and Printed Books
from Respected Authors at a Great
Price!</description>
  </image>

  <textInput>
  <title>Search SpiderWorks</title>
  <description>Search the SpiderWorks Web
Site</description>
  <name>q</name>
  <link>http://www.spiderworks.com/search.pl</link>
  </textInput>

  <item>
    <title>SpiderWorks Releases Danny Goodman's New
Dashboard Book</title>

<link>http://www.spiderworks.com/books/dashboard.php
</link>
    <description>World-renowned JavaScript and
Dynamic HTML expert, Danny Goodman, shows you how
to build rock-solid, professional Dashboard widgets
for Mac OS X Tiger in his new book, Mac OS X
Technology Guide to Dashboard. Includes exclusive
widget debugging tool, The Evaluator! Available as
an eBook and printed edition at
SpiderWorks.com</description>
    <category
domain="http://www.spiderworks.com/topics/tiger.php"
>Mac OS X Technology Guides</category>
    <source
url="http://feeds.feedburner.com/spiderworks">Spider
Works</source>
    <author>Dave Wooldridge</author>
    <pubDate>Tue, 05 Jul 2005 2:37:01 GMT</pubDate>

<comments>http://spiderworks.blogspot.com/2005/07/da
shboard_comments.html</comments>
    <guid
isPermaLink="true">http://spiderworks.blogspot.com/2
005/07/dashboard.html</guid>
  </item>

  <item>
    <title>Spiderworks Interview with Ben
Waldie</title>

<link>http://www.spiderworks.com/books/automator.php
</link>
    <description>SpiderWorks recently sat down with
author Ben Waldie to discuss his new book, Mac OS X
Technology Guide to Automator. Listen to the full
interview online.</description>
    <enclosure
url="http://www.spiderworks.com/audio/waldie.mp3"
length="243108" type="audio/mpeg"/>
    <category
domain="http://www.spiderworks.com/topics/tiger.php"
>Mac OS X Technology Guides</category>
    <source
url="http://feeds.feedburner.com/spiderworks">Spider
Works</source>
    <author>Dave Wooldridge</author>
    <pubDate>Mon, 02 May 2005 0:00:01 GMT</pubDate>
<comments>http://spiderworks.blogspot.com/2005/05/au
tomator_comments.html</comments>
    <guid
isPermaLink="true">http://spiderworks.blogspot.com/2
005/05/automator.html</guid>
  </item>

</channel>
</rss>
```

At first glance, **Listing 1** looks like a typical XML document, but what sets it apart as RSS is the way the XML data is structured. The root element is, of course, the `rss` tag. Nested within that tag is the `channel` tag. It's within the `channel` tag that the meat of the document is stored. **Figure 2** breaks down the main ingredients of this RSS example into groups, providing you with an easy way to visualize the XML code in **Listing 1**.

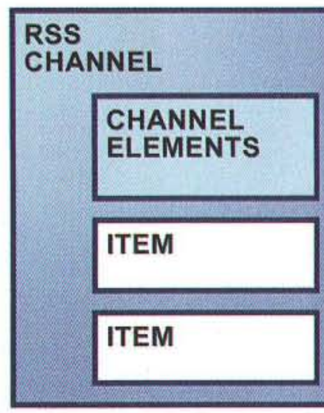


Figure 2. The RSS example broken down into basic groups of elements.

The groups within the `channel` tag consist of two groups of data: (1) channel elements that describe details about the RSS feed itself, and (2) items that hold your news stories, podcast audio tracks, etc. While **Listing 1** and **Figure 2** only include two items for example purposes, you can add as many items as you want to your own RSS feed.

Defining Your RSS Feed

As shown in **Figure 2**, before your actual news items are listed, your XML needs to include some basic information about the RSS feed itself. These tags are called channel elements and include important information like the title of your feed, the content's copyright, the date the feed was last updated, etc. Only a few of these elements are required, but the more information you provide, the more efficiently the

receiving RSS readers (known as aggregators) can handle and process your data. For example, including optional tags like `lastBuildDate` and `ttl` can help relieve the server load from your feed being requested unnecessarily since those tags specify when the feed was last updated and how long the data should be cached (stored temporarily) before refreshing with a new HTTP request of the feed.

While all of the channel elements are defined here, please refer to the code examples in **Listing 1** for the proper XML syntax of these tags.

title

REQUIRED. The name of your feed, which is usually the same name of your blog or web site that's related to your feed.

link

REQUIRED. The URL of your blog or web site (not the URL of your feed).

description

REQUIRED. A very brief phrase or sentence that describes your feed's overall content.

language

Optional. The language that the feed is written in. For list of possible language codes, please refer to: http://blogs.law.harvard.edu/tech/stories/storyReader$15

copyright

Optional. The copyright notice for the feed's content. Do not use the actual copyright symbol since that special character may not display properly in RSS readers.

managingEditor

Optional. The e-mail address of the editor of the content. This is not necessarily the author of the content, since the content may have come from multiple sources, but this is the person who is managing the feed.

webMaster

Optional. The e-mail address of the webmaster who oversees all technical issues related to the feed.

rating

Optional. The PICS rating of the feed, which helps adults control what online content is accessible by children. This tag is rarely used, but for more information on PICS, please visit: http://www.w3.org/PICS/

pubDate

Optional. The publication date of the feed, which states the earliest date that the content can be publicly displayed. Most aggregators ignore this tag and instead focus on the lastBuildDate tag. The date should be formatted to

conform to RFC 822, which can be found at: http://asg.web.cmu.edu/rfc/rfc822.html

lastBuildDate

Optional. The date that the feed was last updated. This is often one of the first tags that aggregators check to see if any new content was been added or updated since the last time the feed was requested. Like `pubDate`, this tag's date should be formatted to conform to RFC 822.

category

Optional. If your blog or web site organizes blog entries and articles into specific categories, then this tag may help aggregators to categorize items accordingly. Unfortunately, there is no standard cataloging system, so often this tag only proves useful for your own site needs. The `domain` attribute typically refers to a URL for that category online, but if your site does not include unique web pages for each category, then you may want to just link to your home page.

docs

Optional. This tag should link to the official RSS specification online. As an RSS 2.0 feed, this tag should point to: http://blogs.law.harvard.edu/tech/rss

generator

Optional. If you used feed generator software to create your feed, then the application would give itself credit in this tag. For example, if you used FeedForAll to generate your feed, then this tag may read:

```
<generator>FeedForAll</generator>
```

skipDays

Optional. This tag informs aggregators that the feed should not be read on certain days. Within the `skipDays` tag is a nested `day` sub-element, so that you can include more than one day within `skipDays`. Acceptable `day` values are: Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, or Sunday. For example, to list both Sunday and Monday as days to skip, you would use the following XML syntax:

```
<skipDays><day>Sunday</day><day>Monday</day></skipDa
ys>
```

skipHours

Optional. This tag informs aggregators that the feed should not be read during certain hours. The `skipHours` tag syntax works exactly like the `skipDays` tag, except that the sub-element is `hour` instead of `day`. An acceptable `hour` value is any whole number between 0 and 23. Like `skipDays` and its `day` sub-element, `skipHours` can include multiple `hour` sub-elements.

ttl

Optional. This tag represents the "time to live" with the value being in minutes. It tells aggregators how long the feed content should be cached on their end before refreshing with a new HTTP request of the feed. For example, if you use 60 as the value, then aggregators will know that they need to wait 60 minutes before requesting a fresh copy of the feed. This can help alleviate some of your server load since it will decrease the number of redundant feed requests.

cloud

Optional. This is probably the most rarely used and most confusing tag in the RSS 2.0 specification. Many developers who encounter this tag either don't understand how it works or have not figured out how to best utilize it. This tag represents a lightweight publish and subscribe feature that includes five essential attributes: `domain`, `port`, `path`, `registerProcedure`, and `protocol`. It is a way for RSS to leverage the power of web services like SOAP or XML-RPC to conserve server bandwidth, allowing aggregators to register to be automatically notified of any updates made to the feed. If you're not using SOAP or XML-RPC web services on your server, then you won't have a need for this tag. For more information on using SOAP with the optional `cloud` tag, visit: http://blogs.law.harvard.edu/tech/soapMeetsRss

image

Optional. You may have noticed that some aggregators display a logo or badge from the feed they are displaying. They get this data from the `image` element, which includes six sub-elements: `title`, `url`, `link`, `description`, `width`, and `height`. For example, if you publish a sports news RSS feed called "Primo Sports Plus" which has its own unique logo, you can supply aggregators with that logo to help brand your syndicated content. There's no guarantee that aggregators will use the logo, but the extra marketing potential makes it worth including. `title` represents the ALT tag if the image is rendered in HTML and `link` represents the URL of your site if someone clicks on the image. The `url` attribute should be the direct URL to the actual image file itself. The image's `description` value is usually the same as your feed's `description` tag. It's important to note that the maximum image `width` value is 144 (defaults to 88 if unspecified) and the maximum image `height` value is 400 (defaults to 31 if unspecified). See **Listing 1** for the XML code syntax of the `image` tag and its nested sub-elements.

textInput

Optional. This rarely used tag is ignored by most aggregators. It provides a way for aggregators to submit a

search (if your site has a search engine) or submit user comments (if your blog accepts user input). If you decide to include this tag in your feed, it requires four sub-elements: `title`, `description`, `name`, and `link`. `title` is the name of the submit button for your text input box. `description` provides user instructions for the text input box. `name` is the form name of the text input box. `link` is the URL of your server-side script (such as PHP or Perl) that should process these text input submissions. See **Listing 1** for the XML code syntax of the `textInput` tag and its nested sub-elements.

The Anatomy of RSS Feed Items

After you've defined your channel elements within the `channel` tag, it's now time to add your actual content items. These are the items that are displayed by aggregators as news stories, blog entries, podcast items, etc. (depending on what content you wish to include in your feed). Each item is encapsulated in its own `item` tag (that is nested within the `channel` tag below the channel elements). See **Listing 1** for the XML code syntax of the item tag and its nested sub-elements. The sub-elements that describe an item's content are defined here.

While most feed items include a link back to the full online version of the article or blog entry, a feed item does not require a link if you wish to include all of the content in its `description` sub-element. In fact, none of the `item` sub-elements are required as long as you include at least the item's `title` or `description`.

title

Optional. This is the title of the item. Although it's optional, most aggregators look for this item sub-element, so it's highly recommended to include it.

link

Optional. This is the URL to the web page version of the item on your web site or blog.

description

Optional. This is the description of the item. For news stories and blog entries, it's your choice to include the entire text or only a summary. If you only include a summary, then be sure to also include the `link` sub-element in your item, so that users can click-through to your site to read the entire story. For podcasts, the `description` sub-element usually holds text information about the song track and artist. Although it's optional, most aggregators look for this item sub-element, so it's highly recommended to include it.

enclosure

Optional. This sub-element defines a multimedia file. If your feed is a podcast, then the `enclosure` sub-element of each item would identify the related audio file. It requires three attributes: `url`, `length`, and `type`. The `url` attribute should be the direct URL to the actual media file itself. The value of `length` should be the file size of the media object in bytes. The value of `type` should refer to the media file's MIME type. See **Listing 1** for the XML code syntax of the `enclosure` sub-element and its attributes.

category

Optional. This sub-element works exactly like the channel's `category` tag, except that it defines a unique category for the individual item. Like the channel's `category` tag, there is no standard cataloging system, so often this tag only proves useful for your own site needs.

source

Optional. This should name the source of the item's content if you are not the original author. For example, if the content is from MacTech's RSS feed, you would list "MacTech" as the `source` value and the URL for MacTech's RSS feed as the value of the `url` attribute.

author

Optional. The e-mail address of the author of the item's content. For example, if the `source` is credited to MacTech's RSS feed and David Sobsey is the author of the piece, then list David Sobsey's e-mail address as the value of this sub-element.

pubDate

Optional. This is the publication date of the item. Like the channel's `pubDate` and `lastBuildDate` tags, this sub-element's date should be formatted to conform to RFC 822.

comments

Optional. This is the URL for the item's related web page of user comments. This sub-element is usually only relevant for blogs that allow web-based user comments.

guid

Optional. This is an interesting sub-element that is used as a unique identifier for the item. There is no set convention for how this sub-element should be used, but most aggregators expect it to be a unique URL string that no other item can have, making it a valid item ID. For news stories and blog entries, this works great since they would have their own unique URLs, but if the item's URL is referred to more than once in your feed, then it cannot be used as the unique identifier here. For a unique URL that will always be available for viewing online, you should include the `isPermaLink="true"` attribute.

Moving Forward

Now that you've stepped through the entire RSS 2.0 specification, you're ready to put RSS to good use. If you're only interested in building your own RSS feeds, then check out the Resources section of this article for links to Mac-

compatible feed generator applications that can help streamline the creation process. If you're comfortable working with XML, you can also use a standard text editor like BBEdit (http://www.barebones.com/) or an XML editor like <oXygen/> (http://www.oxygenxml.com/) to roll your own RSS code. To ensure that your feed adheres to the RSS specification and does not include any errors, always test your feed with one of the many online validators (see the Resources section for some helpful URLs).

If you're interested in parsing RSS in your own software projects or syndicating third-party feeds on your web site, the Resources section also includes links to a few of the most popular RSS parsers for various programming languages. When developing your own RSS-savvy application or web site, it's important to assume that all RSS feeds are invalid until proven otherwise. Because many of the feeds out there are hand-coded, they often contain the wrong text encoding attribute in the `xml` tag and/or just bad XML, so it's a good idea to include a lot of error handling in your own parsing code to safeguard your users from problems. Defensive programming is definitely the name of the game here. It's also recommended to support caching in your RSS applications. If third-parties are kind enough to offer you free content for syndication, be considerate of their server bandwidth by caching the feed content temporarily on your end and only retrieving a fresh feed at timed intervals.

For those of you who want more out of RSS, the 2.0 specification allows you to extend RSS with namespace-defined modules. If you're not familiar with namespaces, they are part of the XML specification that RSS 2.0 supports as an easy way to extend the feed format for custom purposes. For more information on using namespaces to extend RSS 2.0, visit: http://www.reallysimplesyndication.com/howToExtendRss

And last, but not least… once you have your own RSS feed available on your site, it's easy enough to add a link to it on your home page using one of the popular "chicklet" buttons shown in **Figure 1**, but how do you get Safari and Firefox to automatically recognize your feed (see **Figure 3**) with that dynamic feed icon in the location bar and status bar respectively?

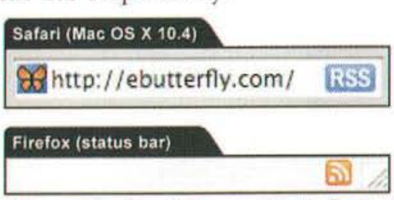


Figure 3. Dynamic feed icons in Safari and Firefox.

The answer is actually quite simple, requiring only a single line of HTML code. Nested within the `head` tag of your web pages, add the following meta tag with the `href` attribute set as the direct URL to your RSS feed:

```
<link rel="alternate" type="application/rss+xml"
title="RSS" href="http://yoursite.com/feed.rss">
```

So what are you waiting for? Now that you've gained some insight on how RSS 2.0 works, start taking advantage of one of the hottest new Internet technologies that can greatly increase exposure for your products and services, and drive additional traffic to your web site or blog.

Resources

The following list is by no means comprehensive, but should serve as a good starting point for learning more about RSS 2.0 online.

RSS Readers/Browsers for Mac OS X

NetNewsWire: http://ranchero.com/netnewswire/
Radio Userland: http://radio.userland.com/
NewsFire: http://www.newsfirerss.com/
Apple Safari: http://www.apple.com/macosx/features/safari/
Firefox: http://www.getfirefox.com/

RSS Feed Generators

FeedForAll: http://www.feedforall.com/
ListGarden: http://softwaregarden.com/products/listgarden/
pReSS: http://www.mizog.com/productinfo/press/
OrangeBox: http://www.globalsyndication.com/orangebox-for-macintosh
Feeder: http://www.reinventedsoftware.com/feeder/

Online RSS Validators

Userland Validator: http://rss.scripting.com/
Feed Validator: http://feedvalidator.org/

RSS Parsers

MagpieRSS for PHP: http://magpierss.sourceforge.net/
CaRP for PHP: http://www.geckotribe.com/rss/carp/
Universal Feed Parser for Python: http://feedparser.org/
RSS Class for Cocoa: http://ranchero.com/cocoa/rss/
WSL-Feed for REALbasic:
http://www.ebutterfly.com/rb/webservices.asp

Documentation

RSS 2.0 Specification: http://blogs.law.harvard.edu/tech/rss

Books

Developing Feeds with RSS and Atom by Ben Hammersley (O'Reilly):
http://www.oreilly.com/catalog/deveoprssatom/

Beginning RSS and Atom Programming by Danny Ayers and Andrew Watt (Wrox):
http://www.wrox.com/remtitle.cgi?isbn=0764579169

About The Author

Dave Wooldridge is the founder of Electric Butterfly (www.ebutterfly.com), the developer of the Web Services Library for REALbasic and the award-winning HelpLogic. He is also co-founder of the new eBook publisher, SpiderWorks (www.spiderworks.com).

NAGIOS ON OS X, PART 1

INSTALLING AND SETTING UP ONE OF THE BEST NET MONITORING TOOLS AROUND

If you read my website, http://www.bynkii.com/, you may know I have talked about installing Nagios in the past. For a while now I've wanted to write an updated, in-depth piece on this subject, and felt now was time to do so, and of course, only in MacTech! I've managed to simplify the install quite a bit, thanks to the hard work of the folks at DarwinPorts, http://darwinports.org.

Welcome

This article is going to have a few sections. First, we'll get DarwinPorts installed and configured. Then we'll use it to install the backend components Nagios needs. Third, we'll do the actual Nagios installation, and the plugin installation. Then we'll go over some of the basics of configuring Nagios. Note that this applies to the current version, which is 1.2. Version 2.0 is in final beta, but I prefer to wait until that's done before installing it on production systems

We should also talk a bit about what Nagios is. Nagios is a network monitoring tool. It allows you to monitor services running on various hosts on your network. For example, you can monitor your switches to ensure they're working, or you can monitor various critical server processes on an Xserve, like the KDC process, AFP processes, etc. You can, with various plugins from the Nagios Exchange site, at http://www.nagiosexchange.org/ monitor internal counters on Windows servers too. If you're skilled with Perl, or any one of dozens of programming languages, you can write your own plugins.

Darwinports

DarwinPorts, like Fink, or any other of a dozen ports management systems is a way to make installing and configuring software easier. If you compile and install open source software manually, you have to figure out all the various dependencies and configuration issues yourself. This is not hard per se, but it is tedious. Ports managers, like DarwinPorts manage this for you. If the DarwinPorts repository has the software you want, you install it, and it handles all the dependencies for you. While, like everything else, there is any number of religious wars about ports systems, I like DarwinPorts, because it does the job well for me. If you like Fink better, great! They're both really good systems.

To avoid modifying Apple–supplied directories, DarwinPorts lives in `/opt/local/`. If you install on Mac OS X Server, then you'll already have `/opt`: it's where Apache2 is installed. The advantage to this is that it makes uninstalling DarwinPorts dead simple. Remove `/opt/local` and all its contents, and DarwinPorts is gone.

Since DarwinPorts doesn't have a Nagios port, we can't just use it for the full install, but we can use it, and some other tools for the support libraries and applications Nagios needs. However, the first thing we have to do is install DarwinPorts itself. The easiest way to do this is to get the disk image with the installer from DarwinPorts, currently at http://darwinports.org/downloads/DarwinPorts-1.1.dmg. Download

the disk image, and run the installer. Once you're done with that, we're going to be doing the rest of our work in Terminal.

To make life simpler, we're going to set up a .profile in our home directory that will make using DarwinPorts easier. While you don't have to do this, if you have multiple versions of software installed in various places, a .profile file will make your life a lot simpler. My .profile is a bit simplistic, but it works for my needs. In any text editor that will allow you to create .files, (pico, vi, emacs, TextWrangler, SubEthaEdit, BBEdit, they're all good) create a file with the following entries, and save it in the root of your home directory as .profile:

```
PATH="/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/share
/aclocal:/opt/local:/opt/local/bin:/opt/local/lib:/o
pt/local/sbin"

export PATH

PKG_CONFIG_PATH="/opt/local/lib/pkgconfig"

export PKG_CONFIG_PATH

MANPATH="/usr/share/man/:/usr/X11R6/man/:/opt/local/
share/man"
export MANPATH

INFOPATH="/opt/local/share/info"
```

The path statement ensures that when you enter a command, that the `/opt/local` tree is used automatically. It also leaves the standard paths for `/bin`, `/sbin`, `/usr/bin`, `/usr/sbin`, etc. The PKG_CONFIG_PATH is used to ensure that software you install under your login can find pkg-config, a tool used to make compiling libraries and applications easier. The MANPATH statement makes finding man pages in the `/opt/local/` tree easier. The INFOPATH variable is used by things like gettext, which figures into a lot of open source packages. If you already have your own .profile, then just add the `/opt/local` information if you wish. I rely heavily on DarwinPorts, so for me, having this .profile setup is a real timesaver. If you don't use DarwinPorts a lot, you may not wish to modify your .profile.

So, once you're done setting up, (or not) your .profile, we're ready to go. If you did modify your .profile from within the Terminal, you'll want to read that file so your environment is set up. Just run

```
source ~/.profile
```

to set up your environment with the new information from .profile. Now we need to make sure DarwinPorts is up to date. To do this, run:

```
sudo port selfupdate
```

and wait a few minutes. If there is any updating to do, you'll see some short status messages on the screen, and if all goes well, you'll get the "selfupdate done!" message, and you're set. If you want to see ***all*** the status messages for a selfupdate, then run:

```
sudo port -d selfupdate
```

which enables debug mode for that command, you'll get all the status messages. Below, I have the two screenshots for the same command on an updated system to give you an idea of the difference the –d makes.

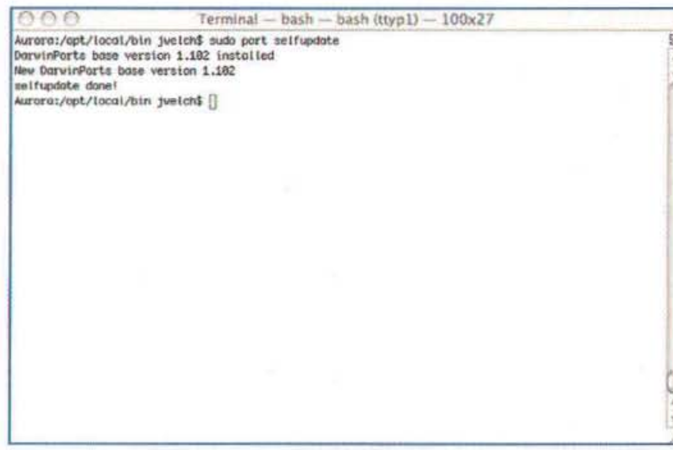


Figure 1: Normal selfupdate

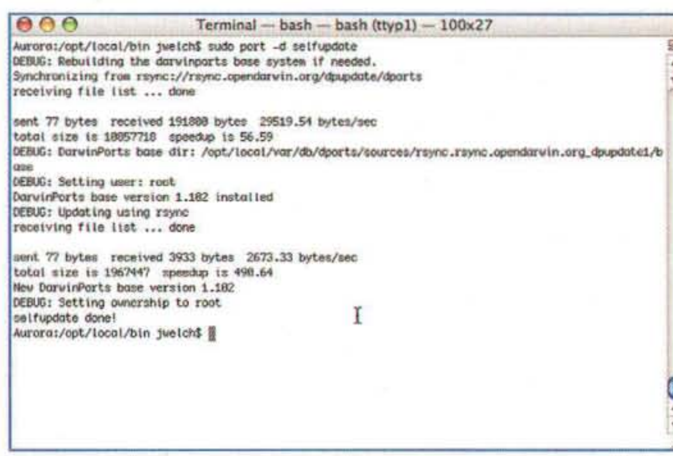


Figure 2: Debug selfupdate

I tend to only use the –d option if I'm having problems, so I can better see where the problem is occurring. I like to run the selfupdate option at the beginning of any install session, so I know I'm starting with the most current version of DarwinPorts.

Once that's done, we install the various background libraries that Nagios needs using

DarwinPorts and the install switch:

```
sudo port install zlib
sudo port install libpng
sudo port install jpeg
sudo port install gd2
```

Once all of those are done, congrats, you've installed the support libraries for Nagios. Note that there are ***tons*** of ports in DarwinPorts, so this is only the barest smidgen of what you can get with DarwinPorts.

We'll need to install at least one particular Perl package for Nagios, namely Net::SNMP. To do this, run:

```
sudo cpan -i Net::SNMP
```

If you've never run `cpan` before, you'll have to run through some configuration steps first. They're pretty easy to follow, and if you're unsure of what to do in a given step, take the defaults. There's even an option to let `cpan` try to autoconfigure itself. I do it manually, but if autoconfigure works for you, so much the better. Using `cpan` is much like using DarwinPorts, only just for Perl. If you're trying to install a Perl module that has dependencies that you don't have installed, `cpan` will prompt you to install those as well. (If you've only ever hear horror stories about downloading and installing open source software, I'm here to tell you that thanks to a lot of hard work by not a lot of people, that process has gotten much easier, and thanks to tools like `cpan` and DarwinPorts, should no longer be thought of as scary to the uninitiated.)

Installing Nagios

Now that we have the support libraries installed, it's time to set up for Nagios. We'll need to create the directory that Nagios lives in, /usr/local/Nagios. Next we have to create the users and groups that Nagios will use. (Security note: ***All*** of this assumes that you have a proper firewall and other protective measures in place. I would ***highly*** recommend that you don't install Nagios on a system directly exposed to the public Internet. A lot of the protocols that Nagios uses, like SNMP v2c are not encrypted or terribly secure, and having a Nagios box exposed to the public Internet will potentially create real problems for you.)

You'll need to create at least a Nagios user and a Nagios group. Note that the nagios user does not need a login shell, just that the account works. Regardless of how you create the nagios user account, (System Preferences, NetInfo Manager, Workgroup Manager, command line), you want to make it a local machine account ***only***, and you want to set the login shell to `/usr/bin/false`. Don't give the account a password. You'll also need to create a nagios group. The Nagios group needs two members, nagios and www. (Adding www to the nagios group makes setting up the integration between Apache and Nagios much easier.) Once you've created the user and group for Nagios, then we're ready to download the Nagios source code files.

Go to the Nagios download site, http://www.nagios.org/download/, and download the 1.2 version of the Nagios tarball, and the most recent version of the Nagios plugin tarball, 1.4.2 as of this writing. Save them both to whereever you want, I ususally save them to my desktop. Unzip and untar the Nagios source code files. In the Terminal, cd to the nagios-1.2 directory. The first thing we have to do is run the configure utility so that we can compile and build the code correctly. As we're using some non-standard (for Nagios) library locations, we'll need to tell it where to find things. We also need tell Nagios where its own base directory is going to be, where the CGI directory it will use is going to be, and where the base web root directory it's going to use is. For this article, I'm using:

```
./configure —with-gd-lib=/opt/local/lib —with-gd-
inc=/opt/local/include —prefix=/usr/local/nagios —
with-cgiurl=/cgi-bin —with-htmlurl=/
```

Make sure you don't have any spaces between the = and the leading / of the paths in the configure command, and let it rip. When you get done, you should not have seen any warning s or errors during the configure, and you should see a status screen like the one below:

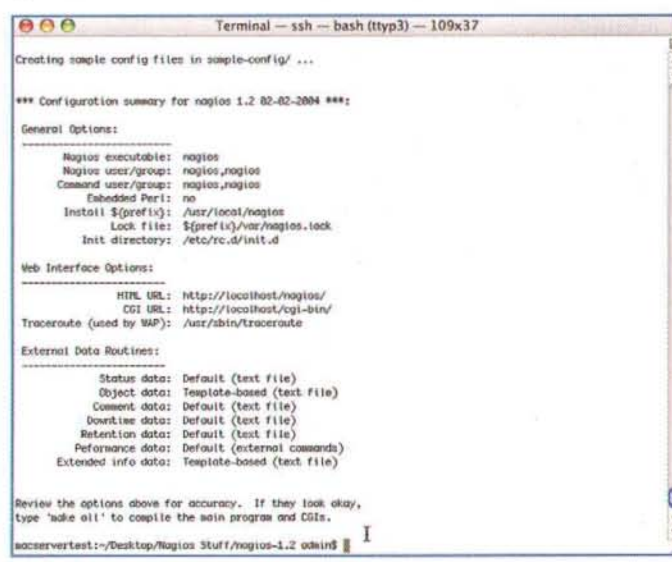


Figure 3: Completed configure screen

You'll want to save this screen somewhere, in case you aren't sure where things are. With this setting, Nagios is going to use the defaults for things, so we'll have to change a few things. First, it assumes that you're going to have all your web files originating from /nagios in the URLs. We'll fix that later. The cgi-bin directory is the system default one, or /Library/WebServer/CGI-Executables. That we'll leave alone.

Assuming your configure went okay, we're going to run the various Nagios make and install commands:

```
make all
```

If make all completes successfully, you'll see a status screen like the one below. We're going to run the other make commands with the exception of make install-init, since Mac OS X doesn't handle startup items like that, and we'll want to test a bit before we make Nagios a startup item.

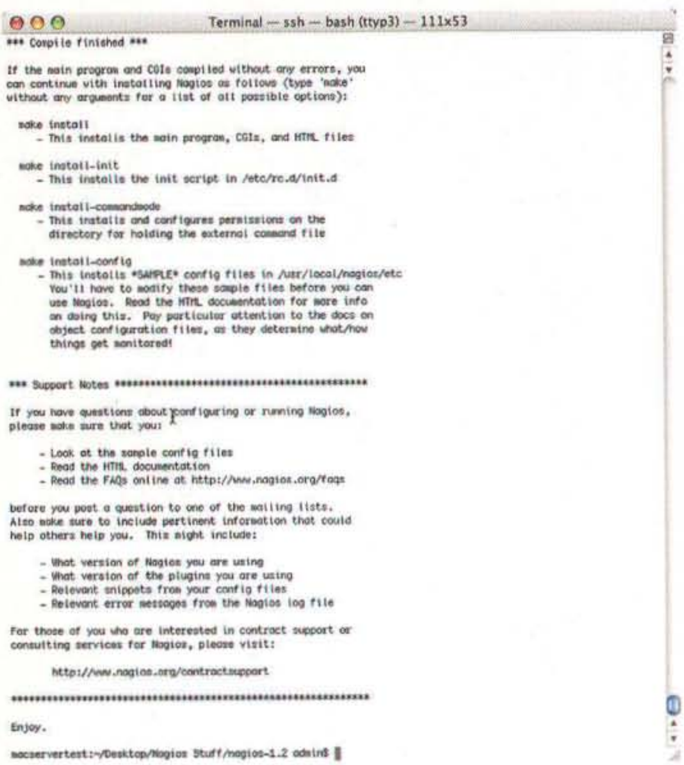


Figure 4: Completed make all screen

```
sudo make install
```

If the make install goes well, you'll get another status screen telling you so.

```
sudo make install -commandmode
```

This command sets up the external command directory for you, important if you want to extend Nagios beyond the basics

```
sudo make install-config
```

This sets up the sample configuration files, which we'll go into later. Next is to configure and install the plugins. As with Nagios, unzip and untar the source files for the Nagios plugins. Change directory into the nagios-plugins directory and run `./configure`. If there are no errors, run `make`, then `sudo make install`.

Initial Post - Install Setup

So Nagios is installed, the plugins are installed, we just start and go, right? Well, not so fast. First, we have to make sure that the web server side of things is set, since Nagios uses a web interface to show you what its monitoring. If you installed Nagios on Mac OS X Server, then you just use Server Admin to point the web server root at /usr/local/nagios/share. You'll want to make sure you have CGI Execution enabled for the site, and that you have the necessary Apache modules enabled to allow you to run Perl and other language CGIs on the site. If you are using Mac OS X instead of Mac OS X Server, you'll have to edit the necessary Apache config files by hand to do this. Oh yes, make sure that evil Performance Cache is turned off.

Next, we copy the Nagios CGIs into /Library/WebServer/CGI-Executables/. The CGIs can be found in /usr/local/nagios/sbin. You want to make sure that at least the group for the CGIs is set to www, and that the permissions are set so that the owner and group can execute. (***I'll leave world executable rights up to your particular needs and security posture***).

The next step is to ensure that Nagios knows where its various web files really are. To do this, we want to open up side.html in /usr/local/nagios/share, and make sure that all the references to CGIs look like this: `href="/cgi-bin/statuswrl.cgi` If they do, then you're all set there.

If you've got everything pointed right, then we can test the basic operation, aka "can we see the Nagios Home Page". Just point your browser at the Nagios URL, and if it's all set up correctly, you'll see:

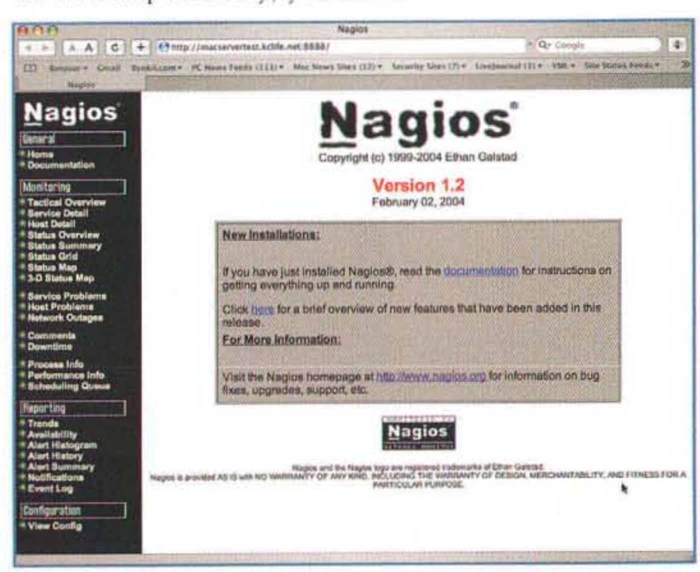


Figure 5: The Nagios Home Page

Note that nothing works yet, and in fact, the Nagios process isn't even running. But at least we

know it can see the home page. So that's something. Next we want to make sure that just anyone cannot see all the information that Nagios can provide. If nothing else, you don't want the entire world, or even your entire company, getting detailed information on your servers.

There are a lot of ways to set up access control to web sites. I'm covering one here, namely using .htaccess files. To do this, we're going to make a couple of changes to httpd.conf, and set up a .htaccess file for locking down access.

```
httpd.conf changes:
```

Note: This a fairly simple way to set this up. There are other ways to do the same thing, and they will work just as well. This one works in my situation.

?Open /etc/httpd/httpd.conf in your text editor of choice (I use BBEdit) and look for the following directive:

```
<Directory "/Library/WebServer/CGI-Executables">
  AllowOverride None?
  Options None?
  Options ExecCGI
  Order allow,deny
  Allow from all
</Directory>
and change it to read:
<Directory "/Library/WebServer/CGI-Executables
  #AllowOverride None
  AllowOverride AuthConfig
  # Options None
  Options ExecCGI
  Order allow,deny
  Allow from all
</Directory>
```

This will allow you to run the CGI's after you have authenticated yourself for the CGIs. To do that, we need to set up a .htaccess file in the cgi-bin directory, and then set up the passwords. Create a file named .htaccess in /Library/WebServer/CGI-Executables/ and set it up thusly:

```
AuthName "Nagios Access"?
AuthType Basic?
AuthUserFile /usr/local/nagios/etc/htpasswd.users
?require valid-user
```

Once you've done that save it and close it. Now, before we create the password file for this, you want to know all the users that will be authenticating. At the very least, you want some sort of catchall Nagios admin user. For our example, we used nagiosadmin, jwelch, and admin

Run the following:

```
sudo htpasswd -c
/usr/local/nagios/etc/htpasswd.users nagiosadmin
```

This will create the user file that our .htaccess file is looking for, and set the first user to nagiosadmin. You'll be asked for the password for this user, set it to what you feel is correct.

To add other users, you run:

```
sudo htpasswd /usr/local/nagios/etc/htpasswd.users
<username>
```

You do this once for each user you want to add. You only need the -c swtich when you are creating the file and first entry.

Then, edit the "use_authentication" line in /usr/local/nagios/etc/cgi.cfg (or cgi.cfg-sample) to read:

```
use_authentication=1
```

If you read that file, you'll also note this is where you set which nagios user can do what. The docs that are a part of Nagios explain this thoroughly, and are available from the "Documentation" link on your Nagios home page, so I don't have to.

Conclusion

I want to stop the article here, because the next part is going to go into the configuration files, which covers a *lot* of detail, and can be pretty dry. Unfortunately, there's no nice or easy way to jump in to that. I ***highly*** encourage anyone who wants to go ahead on their own to do so, but with caution, the config files can get a little odd at times. I'd also encourage any readers who are not going to plow ahead to read the Nagios documentation thoroughly. Nagios has its own language, and the more familiar you are with it, the happier you'll be using it. As always, thanks for reading!

Bibliography and References

DarwinPorts, http://darwinports.org.

CPAN, the Comprehensive Perl Archive Network, http://www.cpan.org/

Nagios, http://www.nagios.org/

Nagios Exchange, the Nagios Plugins and Add Ons Exchange, http://www.nagiosexchange.org/

MT

About The Author

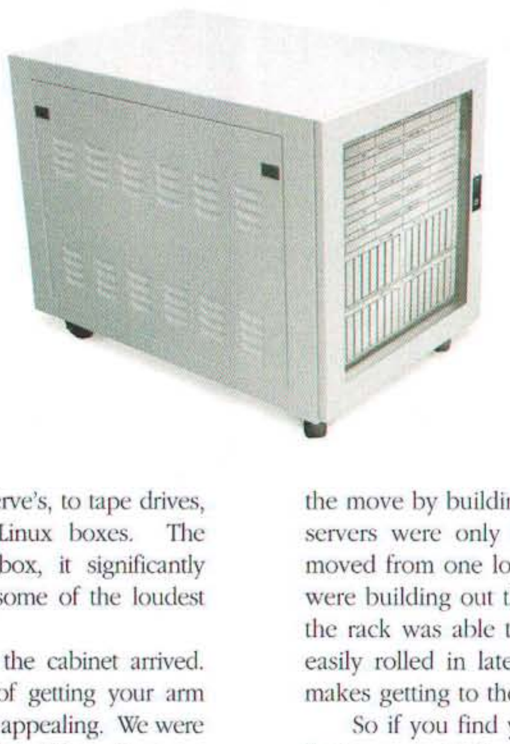
John Welch <jwelch@bynkii.com> is Unix/Open Systems administrator for Kansas City Life Insurance, (http://www.kclife.com/) a Technical Strategist for Provar, (http://www.provar.com/) and the "GeekSpeak" segment producer for Your Mac Life, (http://www.yourmaclife.com/). He has over fifteen years of experience at making Macs work with other computer systems. John specializes in figuring out ways in which to make the Mac do what nobody thinks it can, showing that the Mac is a superior administrative platform, and teaching others how to use it in interesting, if sometimes frightening ways. He also does things that don't involve computertry on occasion, or at least that's the rumor.

XRackPro2

Noise Reduction Rack Mount Cabinet

You walk into the room and are immediately berated by the thrumming and buzzing of machines. You try to stay cool and keep focused on what you're doing, but there's only so much anyone can take. The noise just keeps drumming into your brain making you wonder, "Is this what Chinese water torture is like?" This used to be the server room here at MacTech. And, if you are in the media world, it's not a matter of what is tolerable, it's just not acceptable.

Now things are much different. We have recently become a much less tortured group thanks to the XRackPro. The XRackPro, made by GizMac Accessories, is a rack mount enclosure cabinet designed to reduce noise and add mobility to Apple's Xserve line of servers and RAID systems.



Specifically, it's a rack unit on wheels that provides 12U of industry standard 19" wide rack mount space. Each 1U of rack space is marked.

During the past several months, we've had several different types of rack mount units ... from Xserve's, to tape drives, to even a couple of super loud Linux boxes. The XRackPro not only is a beautiful box, it significantly reduced the noise — making even some of the loudest units tolerable.

We had a few concerns before the cabinet arrived. One was accessibility. The thought of getting your arm stuck inside of the cabinet was not too appealing. We were thus happy to discover that all four sides of the cabinet are removable. So no one lost an arm during installation.

Another concern was making it work to fit our needs. Having several other machines besides an Xserve that we wanted to put in the cabinet we wondered how well everything would fit. So we were happy to find fully adjustable mounting rails which allowed us to configure the cabinet to our own specific needs.

At first, we were wondering whether we would like the "cage nuts" approach that GizMac took, but we grew to see the merits of using cage nuts for mounting. Just make sure in your planning that you have enough of them.

The one thing that we didn't like about the XRackPro was the cable guides that mounted on the rails inside the unit. They came undone far too easily to really be useful. Fortunately, it wasn't particularly important.

This left us with our most pressing concern. Will the noise be reduced? We're happy to say it is greatly reduced. In fact, the louder the device you are mounting, the more impact the XRackPro will have on it in reducing the sound. All of the exterior panels are lined with specially engineered acoustical foam. Each door opening is sealed and all wire grommets come with foam inserts. The manufacturer claims 75% reduction in noise levels. What we can tell you is that there's a noticeable difference between when the front door of the unit is open vs. closed.

There was one other feature that we have found to be a real lifesaver, the ability to have mobility. The cabinet has 4 heavy duty casters. The XRackPro made our recent move MUCH easier at several different points. First, we were able to prep for the move by building the XRack out. On moving day, those servers were only down for about an hour as they were moved from one location to another. In addition, while we were building out the new server room at the new location, the rack was able to temporarily be in one room, and then easily rolled in later. It rolls easily across carpet and also makes getting to the rear of the cabinet a snap.

So if you find yourself being sick and tired of the noise that comes with having several machines running at once, or if you need to have easier access to your Xserve, you should consider an XRackPro.

http://www.xrackpro.com.

$1799 is the list price, with additional accessories available.

MT

By MacTech Staff

MacTech 2005 High-Tech Gift Guide

Gift Giving Goodness

Some of the Best Stuff for the Techies on Your Holiday List

By Michael R. Harvey, Lesa Snider, and Aaron Adams

It's that time of year again, kids, time for our best picks for holiday gift giving. Whether it's Christmas, Chanukah, Festivus, or whatever, the following is what we think are great holiday gift ideas for 2005.

Sonos Digital Music System

When I bought my home a few years ago, I had the opportunity to wire it before the drywall went up. Cat 5 wire went in all over the place, of course, but I also put in a bunch of speaker wire, thinking I would use that to pump music into every room from a central point in the house. Well, most folks don't have the chance to do that. Sonos has a very cool method for getting the same result without having to run a bunch wire all over the floors or tear into the walls.

The Sonos Digital Music System is a hardware/software combination that takes your music collections, wherever they are stored, and lets you play them in any room in the house.

The Hardware

The Sonos Zone Players are the devices you put around the house that receive the music signal. The first one has to be either plugged into a router, or put inline between a computer and router. Hook up the speakers, plug in the power, and you are ready to go. After that, every additional Zone Player you bring online (up to 32 total) can be connected to your Sonos network via Ethernet, or attached through its own Sonosnet secure peer-to-peer wireless network. On the back of each player are RCA jacks and wire terminals for attaching speakers, as well as a four-port 10/100 switch that lets you attach any additional network devices you'd like.

The Sonos Controller is the device you use to manage the Zone Players, and set what songs play on what Zone Controllers. You can have a different song playing from each Zone Player in the house or link them to output the same song. The controller is easy to use: Pick a song, pick a zone, and push play. You can control any zone from anywhere within range of the network. (It could be a great way to freak out someone in the bedroom from the backyard!) Some other features include:

- The 3.5" backlit color LCD screen is easy to read, indoors or out.
- It's splash resistant; all the seams are gasketed.
- It has a built-in motion sensor so it automatically wakes up when you pick it up..

The Software

Run the installer on your Mac (or Windows PC), and it will install the assistant software, guide you through configuring and opening up your network to Sonos, and making your music collections available to the Zone Players. After that, the Sonos Assistant application can manage your rig. With the Assistant, you can name your set up and zones, add/delete additional music collections, and control how the system works and connects. The Sonos system supports MP3, AAC, WAV, and AIFF, among others (although DRM encoded files are not currently supported).

The Sonos Digital Music System is for audiophiles who must have their music follow them throughout the house. This will be a pricey gift. The base system of two Zone Players and one controller will run \$1199 (a system including two sets of high performance bookshelf speakers is \$1499). Additional Zone Players are \$499 each. Go for this and you'll pretty much blow the Christmas budget; though, you will score serious brownie points if you give it to someone instead of keeping it for yourself.

One final note about purchasing a Sonos system: When you do, you get an extra bonus. Because they have entered into an agreement with Get Digital music conversion service, you get a discount on their CD to MP3 conversion service. On an order of 75 or more discs, you get 25 free. A pretty nice deal on a very good service, but we'll tell you more about that later. www.sonos.com

MT

EyeTv and EyeHome

Tivo is a great product. It has a slew of features that make it better than any VCR ever conceived, but it's too easy. It is a consumer device. El Gato has a far geekier solution to get the same functionality, plus a few tricks Tivo can't turn. The combination of the EyeTV 200 and EyeHome units get your TV, Mac, home network, and nerd gene all involved in the digital recording of television content.

The first component of the fun is the EyeTV 200. It's a small box that gets your cable TV signal into your Mac. It has a 124 channel analog TV tuner that can take in any basic (unencoded) cable signal via coax, and put it on your computer screen. It also has composite and S-Video video in ports for hooking up, say, a VCR. With those extra connectors you can move old recorded tapes onto your computer via the EyeTV box. Unlike its predecessor, the EyeTV USB, the 200 connects via—and can be powered by—FireWire. FireWire allows the EyeTV unit to move more data into the computer, making recording of higher quality data possible. The EyeTV 200 turns your computer into a DVR, and records MPEG-2 DVD quality data. Pausing live television, time shifting, and recording shows—it's all there. Plus, EyeTV does things your Tivo cannot. With the included and regularly updated software, you have the ability to do basic editing of your saved TV shows. You can also burn saved shows to VCD or DVD, using Toast. Or, you can export your recorded shows into iDVD, or iMovie. Programming of EyeTV is handled through the TitanTV web site. They integrate seamlessly, and it works very well. In fact, you can set up TitanTV and EyeTV to program shows for recording over the Internet. The EyeTV is, by itself, a great product. It's just fun to use.

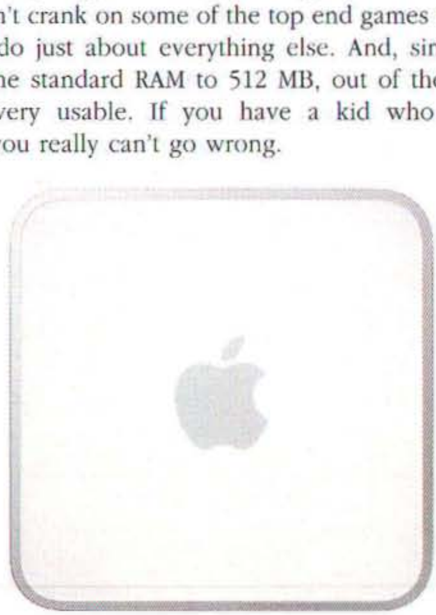
The next component in this patchwork of geekery is the EyeHome. This is the bit that connects your computer's EyeTV archive of shows with your TV, over your home network. Note, however, that you don't necessarily need a wire to put the EyeHome on your network. El Gato has instructions on their web site that describe how to use an Airport Express to put your EyeHome on the LAN wirelessly. EyeHome has a software component that provides the link to the EyeHome box. The box itself connects to the TV via component or optical cables. When it is turned on, and logged into your computer, you get a menu that lets you access the stored EyeTV shows you've recorded. That's not all, though: The EyeHome allows you access to your iTunes library, your iPhoto library, and the contents of your Movies folder. You can play all the clips you have stored on your computer; something Tivo has been promising Mac users for years but has yet to deliver. EyeHome also has Web access, but it works about as well as WebTV, which is to say, not at all. The menu driven interface is okay; it's not the smoothest operation, but it works and you can watch most anything. It plays MPEG 1, 2, and 4, and DivX. It plays a laundry list of audio codecs, as well as displays jpg, png, gif, and bmp image formats. The EyeHome has it's own remote with pretty standard function keys.; it gets the job done. Even better, though, if you've got a Harmony Remote, you can program it to control any of El Gato's products.

Either of these units, by themselves, are a neat addition to the geek's arsenal; particularly the EyeTV. It could be especially useful in, perhaps, a dorm room, where space is at a premium. Both together, though, are a great joy to use and mess with. The folks at El Gato are constantly improving the features and utility of both units and the software. Any person on your holiday list who loves to mess with stuff, or do things the differently, will really enjoy these two items. www.elgato.com

MT

One for the Kids

The Mac Mini is an excellent little computer for kids. Tiny, quiet, and plenty powerful, it is the perfect computer for your youngster's bed or dorm room. Tack on a LCD screen, keyboard, and mouse, and you have a compact unit capable of anything a school kid might need for around $850. It won't crank on some of the top end games out there, but it will do just about everything else. And, since Apple upgraded the standard RAM to 512 MB, out of the box the system is very usable. If you have a kid who needs a computer, you really can't go wrong.



MT

Lift and Separate

For some time now, conventional wisdom has dictated that tilted keyboards are bad. Ergonomically, the angle your wrists sit while typing on a titled keyboard promote repetitive stress injuries. So, since the PowerBook G3, the ability to angle the keyboard of Apple laptop computers has been missing. While that was good for your hands, it had the minus effect of removing the airflow from underneath the laptop. Heat ensued. There have been a few products released to help with this issue, such as the XStand, which happens to be the perfect stocking stuffer. It consists of a collapsible metal frame which lifts a laptop off the table. It is size adjustable to securely hold any sized laptop computer, as well as height adjustable (you can increase the angle). The one issue is that the minimum angle of the XStand is still too great to type on comfortably. It does produce good air flow underneath it, and places the screen at a more comfortable height off the desk for viewing when you plug in a keyboard. This is a really good gift for the PowerBook user in your life. www.xstand.com

MT

Gigs on the Go

Small, USB thumb drives are becoming standard fare on keychains all over the place. They are great and very convenient, but sometimes they lack the capacity to move the big stuff. Having portable drives that can run on bus power is a great alternative, such as the one from Transcend. The StoreJet 1.8 is a USB 2.0 drive that is backwards compatible to 1.1. It's very compact, smaller than a deck of cards, and quite zippy. Format it the right way, and it will plug into both Windows and Mac systems without a hitch. This is great for moving files between platforms, as well as between computers that lack FireWire ports. It comes in 20 GB and 40 GB sizes. www.transcendusa.com

MT

Carry On

There are myriad laptop cases and backpacks out there, and our advice (whenever we review them) is to put your hands on it and try it out before buying. That can be difficult to do when gift buying, so you'll just have to take our word for it on these two. Both of these cases are really good choices for someone who wants a good-sized bag that carries a lot, but just as importantly, looks both cool and professional.

From Brenthaven, a perennial favorite here at MacTech, is the ProFile 12/15 case available from the Apple Store. This $129 case is an excellent combination of size, looks, and luggablity. It is nearly square in shape, a bit thicker than some other shoulder cases you might come across, but very comfortable to carry on

your shoulder. There are numerous pockets to carry any number of cables, accessories, papers, or you name it, a strap built into the side that you can slip over the handles of your roll away luggage, as well as the ultra secure laptop compartment that can handle up to a 15" laptop. Class and utility. This is a perfect choice for the business traveler on your list. www.brenthaven.com

Now, Timbukt2 cases haven't ever really been known for staid, classic looks. Cool and wild are more fitting labels. However, they do have the Laptop Zip Briefcase, which boasts a very classy look while still being cool. This bag, made of ballistic nylon, is cavernous. If they made a 35" laptop, the Zip Briefcase could likely handle it. It has a very comfortable shoulder strap and handles (more comfortable than the Brenthaven), two outside pockets—one perfectly sized to hold plane tickets—and a well-padded interior. The wild comes into play on the colors. The design and look of the Zip Briefcase is very professional, but in olive green or Navy blue, and it becomes a little more cool, something other cases of its ilk lack. It's an excellent option for the young, mobile professional. www.timbuk2.com

MT

Lots of Fun

While there haven't been a lot of games released this year for the Mac, compared to earlier years, the quality of those games has been particularly good. Here are some of the best, gift-worthy picks for your favorite gamers this holiday season.

Let's start right at the top with the very best: Lego Star Wars. Ported over from the console arena, this game is more fun than should be allowed. Played from the third person perspective, you take on roles of over 30 characters from the first three Star Wars movies, from Jango Fett, to Jar Jar Binks, to battle droids. Play through various scenarios of the first three movies as Legos. Game play is very easy. There is very little difficulty to playing, so anyone can quickly get up to speed and start enjoying this game. Having said that, the simplicity of this game is what makes it so enjoyable. You don't have to worry about solving complicated puzzles or

anything like that: You just have fun being a little Lego dude. It's great fun for anyone in the family to play. Bonus items collected through a level help you build a Lego version of a ship from that level (a pod racer for example). Collect Lego bits along the way, as money, and buy add on items. For example, you can give everyone mustaches, or (and this was my favorite) turn all the light sabers purple. Because it began life as a console game, Lego Star Wars plays best using a game controller. The Nyko Ariflow, reviewed earlier this year in MacTech, is one of the better choices. It is very comfortable, has good, easy to use, set-up software, and best of all, has a fan that pushes air through vents in the controller, cooling your hands and keeping them from getting sweaty. www.aspyr.com

Next is Command and Conquer Generals: Zero Hour. The original Command and Conquer game was ported to the Mac, but the sequels haven't been seen on our platform until last year when Aspyr ported the latest incarnation, Generals. Zero Hour is an excellent add-on pack for Generals. It provides new units, buildings, maps, and additional game play that gives you many more hours of fun with this game. If you or yours is a fan of real-time strategy, Command and Conquer is the best there is. www.aspyr.com

Another great strategy game is Rise of Nations. In this universe, you control the powers that make civilization. You begin in ancient times, with a single village, and grow from there, building your civilization through the ages. This game is much like the old Civilization games, but instead of being turn-based, you play this one in real time. Conquest is the main goal, building your armies to take over any of the 18 other civilizations you may face. You can also build Wonders of the World, from which you receive bonus abilities. There are two other game play modes. One is multiplayer, where you can set and manage the parameters of the scenario, and have others join in via LAN or internet. The other is Conquer the World, a series of linked scenarios that take you through time in an effort to take over the world. www.macsoft.com

Then we've got Project Nomads, from Freeverse Software. This game is a third person perspective shooter, essentially, but it's also more. It's got a very interesting story, and one of three characters to play the game with. Each character has his or her own strengths that you leverage to complete missions. The world in which you operate is one where your planet was destroyed, leaving massive islands of rock floating in the ether. It reminds me

a lot of the novel The Integral Trees by Larry Niven. Your mission is to find those who caused the destruction of your world and destroy them. You have the option of LAN or Internet multiplayer gaming action. One of the really cool things about this game is the freedom of movement. Fly from island to island, with no boundaries. It's fun getting to explore outside the confines of your current mission. www.freeverse.com

Fifteen Minutes of Fun

This this day and age, you often don't have time to commit to playing an involved game like the ones above. Sometimes you just to want to kill a few minutes with something fun. GooBall and Jammin' Racer are two excellent choices.

GooBall, from Ambrosia Software, is odd, quirky, and really nice to look at. You are, well, a little ball of goo, with four eyes and a zipper, trapped in a bubble, winding your way through various levels, reaching the finish line as fast as you can. You have the ability to leap over gaps and stick to walls. It makes for some vertigo inducing perspectives. The level variety is astounding and very imaginative. It's a blast to play. www.ambrosiasw.com

Next up is Jammin' Racer from DanLabGames. Here, you compete in various dune buggy races, trying to, obviously, win the race. Each Cup (there are several) has four races which must be won in order to advance. Drive your sand rail in all manner of environments, even on the moon. There are a bunch of customization options for your ride, and you can also go head to head with two player gaming. It's a bunch of fun. www.danlabgames.com

A Little Light Reading

The Unofficial Lego Builder's Guide, by Allan Bedford, is the book for the person in your life who still, to this day, has their very first Lego set, and who still pulls it out for periodic assemblage. This book is not for a seven year old; it's for serious Lego freaks. It shows you, among other things, ways to put Legos together, how to sort and store them, and even gives you a step-by-step for building a space shuttle. Very cool. From No Starch Press.

These next books are some pretty heady stuff. The Apple Training Series, from Peachpit Press, is an Apple Certified series of technical books designed to help you, or yours, pass the exam for the various Apple certifications. Edited by the likes of Owen Linzmayer and Schoun Regan, these volumes are packed with information on supporting Mac OS X, laptops, desktops and servers. These babies are essential additions to the library of any system administrator or techie looking to become Apple certified.

Nice Dreams

Nobody gets enough sleep anymore. There just isn't time, or you can't shut your brain down, and it runs and runs, keeping you awake into the wee hours. Medication can help, but popping pills isn't a long-term solution. There is another way. Pzizz is a very interesting little program that will run on your Mac and generate naps that you listen to on your iPod, helping you get that power nap, or couple of hours of sleep.

The interface looks a bit like a collapsed iTunes window, and using the program is as simple as using iTunes. Clicking on the generate nap button will do just that. You can configure how the nap, or sleep, will be generated from within the Configure Naps window (below). Here you set the length of time, volume, whether or not there is a voice, as well as view the stored configurations you already have set up.

I tried Pzizz on a red-eye flight from L.A. to Boston and it was wonderful. I have trouble sleeping on planes, but I got four good hours with the four, one-hour long sleeps I created. I got to Boston rested and ready to go to work. Compare that to the same flight a year earlier, when I showed up ready to go to bed and couldn't get my act together for a day and half. Running four one-hour long programs on the iPod ate the battery, though. You might want to create smaller sleep files to increase battery life.

Xeric Design, the makers of Pzizz, have the product broken down into two parts: The main part is the program itself and the energizer module. There is also the add-on sleep module, that lets you create the longer sleeps. Pzizz is $39.95, including the nap module, while the sleep module is an additional $19.95. www.xericdesign.com

LOOK FAMILIAR?

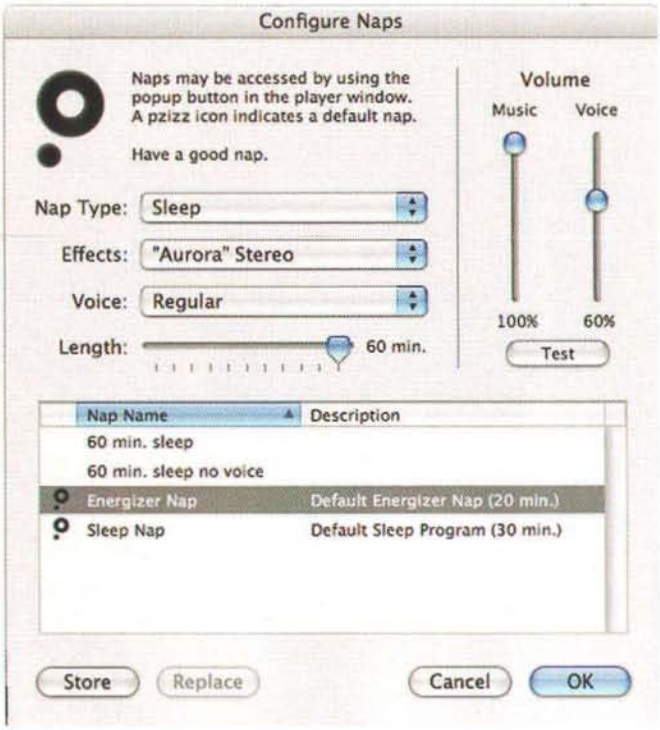
Get Rid of Your Fax Machine!

Receive your faxes directly to your email account as PDF file attachments

Service available in over 150 area codes

To check out special offers for MacTech readers, visit www.MaxEmail.com/MacTech

maxemail®



Musical Digerati

You know you have someone in your life with stacks of CDs and a nearly empty iPod. They, like almost all of us, simply lack the time to put each disc through iTunes and convert it. In the past, the only options for conversion services were college kids in their dorm rooms, and the results were as varied as the quality of the dining commons lunch menu. Now, Get Digital is on the scene to put a professional face on this type of service.

The service itself is very many things: Very fast, very professional, and very good. When you order their service, they send you all the packaging materials you need to get your CDs to them, including a prepaid FedEx label (the level and cost of which you set when you order the service). Once you get your collection to them, it takes mere days to completely digitize your collection into 192kbps MP3 files (or higher, if you'd like), including full, and accurate metadata, which they load onto DVD media or your iPod (or other MP3 device), if you choose to ship it to them. It's then reloaded into the sturdy shipping boxes and sent back to you, also via FedEx. Oh, they send all the discs back to you in the same order you sent them in, making it much easier to get them back into their jewel cases. A very nice touch.

The pricing and options on the service are another very as well: very reasonable. The base price of the service ranges from 99 cents to $1.49 per disc, based on the number of CDs you send them. The

standard return method is DVD, but you have the option of purchasing a Maxtor external drive, with options that begin at 100 GB for $185, and have your music loaded onto that instead. Sending them an iPod to load incurs no additional cost. And, for $1 they will drop a three-ring binder in the shipping box, with a color catalog of your collection, showing cover art, and tracks (the catalog is in alphabetical order).

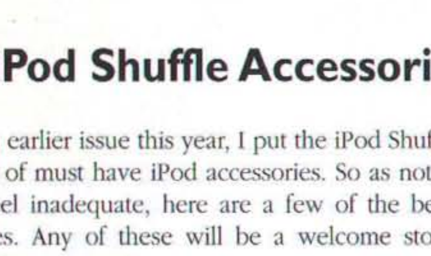
I sent them 69 discs and tried to drop a few zingers in to trip them up (a really old Mozart CD, and an obscure country music disc, among others). They got all the data right and could not find cover art on only two CDs. Not too bad, at all. And don't forget the deal you get with the Sonos system! www.getdigitalinc.com

iPod Shuffle Accessories

In an earlier issue this year, I put the iPod Shuffle at the top of my list of must have iPod accessories. So as not to make the Shuffle feel inadequate, here are a few of the best in Shuffle accessories. Any of these will be a welcome stocking stuffer come Christmas morning.

Belkin Hi-Speed USB 2.0 hub

Now, this hub from Belkin is, in essence, much like every other USB 2.0 hub. It is powered, backward compatible to USB 1.1, and all that other stuff. The one thing that sets it apart is a port perched atop the hub, making it easily accessible for your iPod Shuffle. Getting around to the back of your iMac, or pretty much any other computer, except the G5 tower, is tough, especially when adjacent ports are already in use. With the Belkin hub, it's dead simple to hook up the Shuffle. www.belkin.com



DLO Action Jacket

Exactly 42 seconds after the Shuffle was announced, 15 different companies shipped some form of protective covering for it. And they are all pretty much the same: Varied bits of plastic in various colors. Yawn. The DLO Action Jacket, though, stands out. It's a good pick because of its versatility out of the box. It comes with both a belt clip and an arm band. The padded case is made of neoprene, so it's a nice protector against moisture (like sweat), as well as from bangs and scratches. Pull the case off the belt clip or arm band, and just have it hang around your neck with the standard Apple lanyard. The belt clip is strong, and lets you rotate the Shuffle 180 degrees. The arm band will fit almost any arm thickness, from scrawny to brawny. A very good all-in-one solution. www.everythingipod.com

XtremeMac AirPlay Shuffle

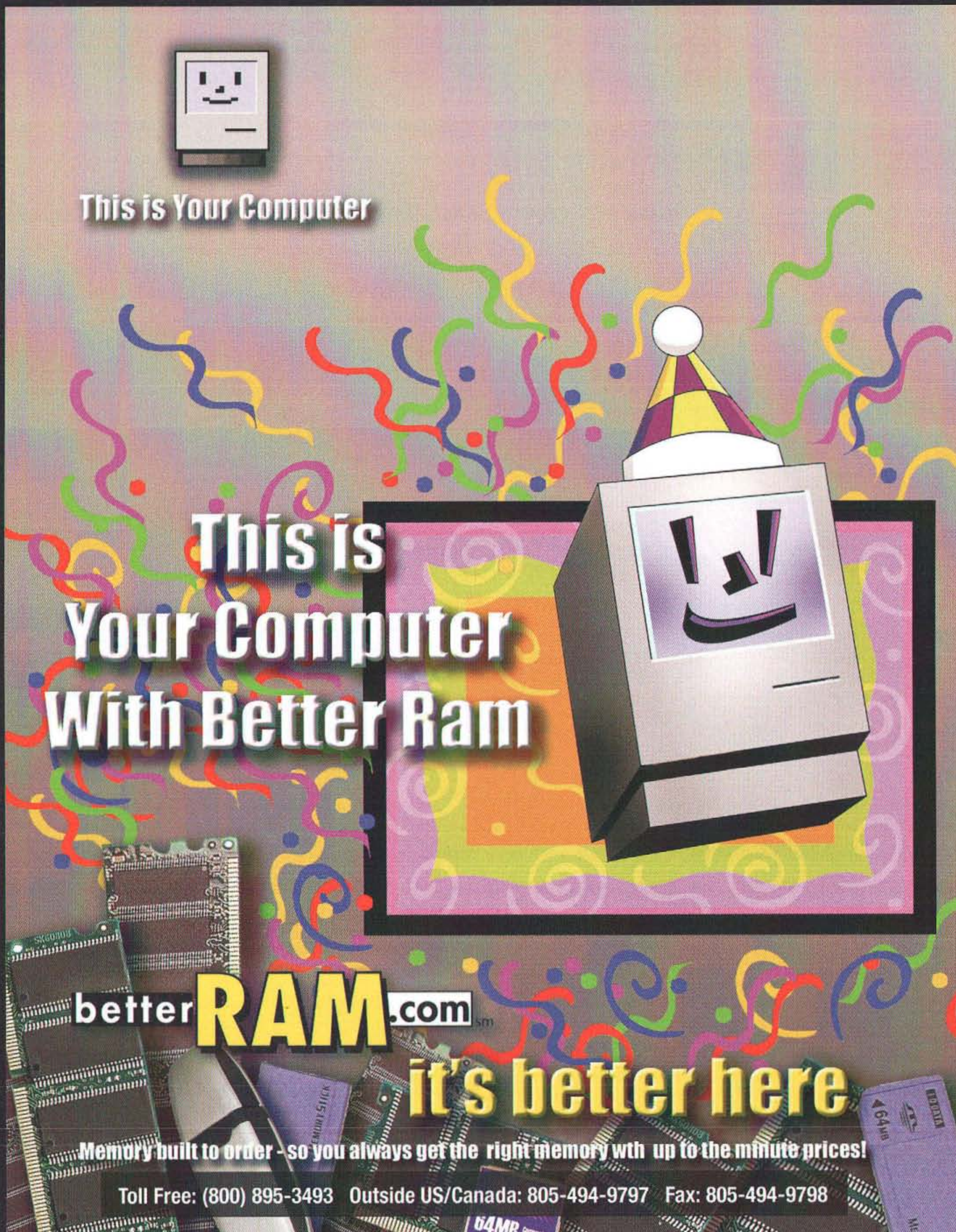
One of my favorite iPod accessories has always been FM transmitters. XtremeMac came out with the first one for the Shuffle. It's a small unit that plugs into the power port in your car which charges the Shuffle, as well as transmits the signal over FM. It has a small LCD screen showing the current frequency it's on. Also, and this is its nicest feature, it has three preset buttons, allowing you to retune it with one press. A good idea considering the numerous distracted driving laws floating about these days. www.xtrememac.com

Look Ma, No Hands

Speaking of hands-free cell calls and distracted driving laws, we had the chance to try out a couple of hands free headsets, either one of which will work great for gift giving, depending on the cell phone.

First up is the Jawbone made by Aliph. This headset uses what Aliph calls Adaptive Technology to cancel outside sound while also picking up your voice more effectively than other hands free units. In testing in noisy environments, the Jawbone worked very well; my voice was clear to the caller and I could hear easily. The ear piece is pretty comfortable, too, albeit a bit heavy. It is easily adjustable to fit into either ear. There is a clip about half way down the wire, which houses the Jawbone's circuitry. The clip also has a button that acts as either an accept/end call button, or mute button, depending on which phone you model you use, as well as an on/off slider to activate or disable the adaptive aspects of the unit. The catch with this unit is the connector: Because the Jawbone interacts with the phone's software and needs to draw power from the phone's battery, it only works with a limited set of Motorola, Nokia, and Sony Ericsson phones that have data ports on the bottom. You will have to check the Aliph web site to make sure a phone is compatible. If it is, it's a good choice. www.aliph.com

For those of us lacking a phone supported by the Jawbone, Shure has an option that is at least as good. Shure makes high-end audio equipment, including head sets and ear buds. Their sound isolating designs work better, in my view, than the best sound canceling technology out there. They have extended this design genius to a hands-free cell phone unit. The QuietSpot Boom headset (model numbers QSHB3 or QSHB4) is a great, light-weight, ear piece, with tiny boom mic that plugs in, out of the box, to the hands-



This is Your Computer
This is
Your Computer
With Better Ram
betterRAM.com
it's better here
Memory built to order - so you always get the right memory wth up to the minute prices!
Toll Free: (800) 895-3493 Outside US/Canada: 805-494-9797 Fax: 805-494-9798

free port on most every phone made. Those few who don't have the proper 2.5 mm jack can get an adapter to make it fit. This unit fits instantly into either ear, as it's held in place by wrapping the wire over your ear. Check the web site for a complete compatibility list. It's very comfortable. The sound, both in and out, is also excellent. The noise canceling boom mic, which eliminates up to 70% of outside noise, picks up your voice very well. The ear plug (the unit comes with an assortment of sleeves to fit any ear canal) does an outstanding job of keeping outside noise out, making it very easy to hear who's calling you. www.shure.com

More Free Sound

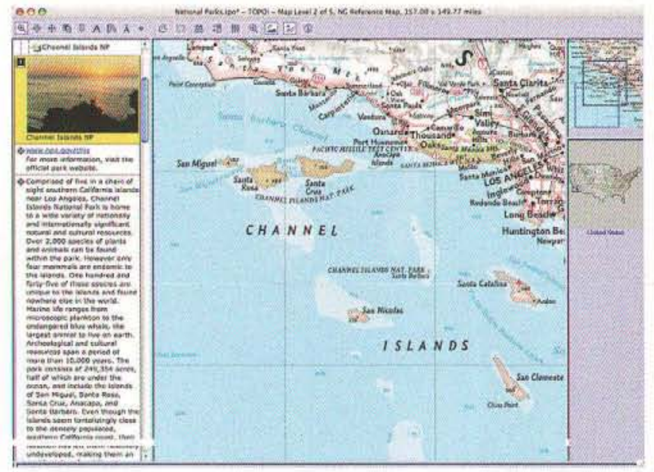
Bluetooth is cool; it makes wires go away. The things that can be done with it have the potential to make wires utterly obsolete. A move in that direction is the iCombi, a wireless headset unit designed to receive signals from, among other things, your iPod.

iCombi isn't the first unit of this type, but it is, so far, the best. The design of the headset is pretty compact. The materials are strong, and the unit is well-built. It has a folding design to make it easier for compact storage. Built-in batteries are charged via USB or wall charger. While it's heavier than most headsets, it's not uncomfortable. You can keep it on for long periods of time.

The headset itself is only half the equation. It links to the other, and more important, half. With a dongle designed to attach and be powered by your iPod, you can listen to your tunes without getting caught up in the wire snaking from your iPod to your ears. It's great if you're active, say running, skiing, or the like. It can also connect to a variety of other devices, and your computer can feed sound to the headset. If your computer doesn't have Bluetooth built-in, iCombi has an optional USB dongle designed to make the link. Also, iCombi has a general audio dongle for connecting to standard audio devices, like a stereo, giving you another way to play tunes wirelessly on the headset. And, just to be clever, the iCombi headset can also act as a hands-free device for your cell phone with its built-in mic. Just pair the headset to your Bluetooth cell phone and you can move, with the press of a button, from listening to music to answering calls. www.icombi.com

Explore

The National Geographic Back Roads Explorer is a map geek's dream come true. Delivered to you on 18 CD-ROMS, this package has detailed topographic maps of the United States, including Alaska and Hawaii. With this application, you can plan



trips, annotate and print out maps, and transfer and view them on either Palm or Pocket PC devices. The CDs are loaded with over 2,000 U.S. Geographical Survey maps, with road and trail overlays, giving you a very detailed overview of any area. Plus, you can use the Back Roads Explorer to program your GPS to get you where you're going, as well as get the coordinates for over one million locations.

The installer comes on Disc 1 and loads up the application, named TOPO, and lower detail maps of the US. You can copy the more detailed maps of each state by copying the directory from the state's CD to the TPO_DATA folder on the hard drive. The interface isn't the prettiest thing you ever saw. You can tell this is a port, but it is usable, and feature complete. If you have someone who is into maps or geography, or would like a way to build really detailed travel routes, or just find interesting things in the places you are heading to, this is a great box to put under the tree. maps.nationalgeographic.com/topo/backroads/

It Came Inside A Midnight Beer

This Jingle Isn't A Bell

Company holiday functions populated by the inane and the insipid. A Christmas Eve spent with distant relatives, unseen since last Christmas, yammering on about the minutiae of their boring personal lives. Mounting credit card debt. The same boring Christmas specials that have been on television for twenty straight years. A freezing Saturday afternoon spent hanging uncooperative lights and decorations. In-laws.

What do all these things have in common?

Beer!

Whether it's being consumed on a silent night at home, at a holiday party, or to take the edge off of awkward social situations, alcohol is part of the holiday ritual for many of us. And what

could be better on a cold winter night than an overstuffed recliner, a crackling fire, and a cold glass of thick, black, bitter stout? I speak, of course, of Guinness, the magic wonder brew that seems, at times, to be the solution to all of life's problems.

Guinness drinkers may have noticed of late a jingle in the can or bottle of their favorite holiday hops. That noise is none other than the Guinness widget, a clever device invented by Alan J. Forage and William J. Byrne, according to the widget's patent, which forms Guinness's characteristic creamy, foamy head once a can or bottle is opened.

What's so special about Guinness that it needs a widget?

Beers are carbonated to different degrees as a consequence of the yeast metabolizing sugars in the fermentation process. Sometimes additional carbon dioxide is dissolved into the brew as it is packaged. The dissolved CO2 in the beer causes the container to become pressurized. Later, when the beer is opened, the pressure is relieved and the CO2 bubbles out of solution to the surface and creates a head on the beer. Guinness is a little different. It contains less CO2 than most beers so nitrogen is added, which works in combination with CO2 to create the smaller bubbles that enable Guinness to have its thick, creamy head. The nitrogen dissolves into the beer, but not very well, so it bubbles out easily when the container is opened. The CO2 in Guinness, however, is more difficult to extract - the beer isn't saturated with enough CO2 for it to naturally bubble up and create a head during decompression.

Enter the widget (no, not that kind of Widget). The widget resembles a ping pong ball about 3 cm in diameter with a small hole 0.61 mm in diameter created by a laser. After the widget is dropped into an empty can, the can is filled with beer and a quick shot of liquid nitrogen is added. The can is immediately sealed and proceeds to pasteurization where it is heated to 60 degrees Celsius for 15 to 20 minutes and the liquid nitrogen in the can boils into gas. The expanding gas creates pressure of about 25 PSI inside the can, causing some of the nitrogen and beer, 10 - 15 ml, to be forced into the hole in the widget. The can and its delicious contents are then distributed to retailers.

Sometime later, the Guinness is purchased and opened. The pressure inside the can equalizes with the ambient pressure outside, and the beer and nitrogen contained inside the widget is forced out. The subsequent jet disturbs the Guinness around it, essentially shaking it from the inside out, and the remaining nitrogen and stubborn CO2 bubble out of solution, rising to the top to create the thick, creamy head Guinness is famous for.

This seemingly simple system took Guinness 20 years and more than 100 failed attempts to perfect.

Widgets are present in Guinness bottles as well as cans. The bottle widget works on a similar principle—compress nitrogen and beer in a plastic container and release the contents when the pressure changes—but is shaped like a rocket instead of a ball. When the bottle is opened, the rocket launches off the bottom, disturbing the beer inside and creating the head. Each time the bottle is tipped back for a drink, the rocket releases a little more foam-producing gas and the head on the bottle is topped off. And the rocket does have fins, not for aerodynamic reasons, but to keep the widget from exiting the bottle where it could be swallowed, hence ruining the taste of the beer.

This holiday season, take a moment to salute the geniuses ("Brilliant!") behind the widget when popping the top of a bottle or can of smooth black. Their contribution to seasonal stress relief is immeasurable

MT

Waterproof Camera Housings

Protect Your Digital Camera From Crashes, Splashes, or 150 Feet of Depth

Yeah, yeah, this is the journal of Macintosh technology. We know. We also know that a large number of our readers are avid camera enthusiasts. Over the years, we have brought you reviews of some of the best digital cameras we have found. To start off this year's gift guide, we thought we would share with you what we've learned about some cameras we have tested, but more importantly, ways to protect them from the elements. The ones you might find at the beach, on the slopes, or even at the bottom of the sea.

Now, Michael is an avid SCUBA diver. He's been flirting with nitrogen narcosis since he was sixteen. Lesa is a super genius when it comes to making your shots look their absolute best. Between the two of them, you will have a very good idea of what cameras and cases you can get, and what you can do with the digital images you capture with them.

Testing Conditions

We figured the best way to stress test the cases we had would be to pick the most extreme conditions we could manage, and that meant diving. We made both warm and cold water dives, and did some shallow water snorkeling, to test the light weight cases. Following are the rigs we used, under what conditions, and what our impressions were.

The Equipment

Nikon

First we tested the sturdy Nikon WP-CP3 (pictured above, back row, right). It is designed for the Nikon Coolpix 4600 and 5600 cameras, both of which are very capable units. You will find, as we go along, that most of these cases have similar features. Clear plastics made of polycarbonate, spring-loaded buttons, slides, and wheels of differing strength (depending on the case's depth rating) to allow you to access the control functions on the camera, and securing latches of varying design.

The Nikon case had a wheel to allow you to change the mode of the camera, and that worried me a bit. I thought it was a weak point which could allow water into the case easier than a button would. It was fine, though. I turned it quite a bit at depth, and not even a drop slipped through. The buttons on the case were, by necessity, clustered tightly together. They were also labeled with small pictograms on the case. Clear icons on a clear case made it hard to figure out which button did what, until you got used to it. Also, because the buttons are so close together, pushing the right one can take a bit of finesse. Don't worry, , I'm not picking on Nikon; almost all the other cases suffer the same problem. It's a result of the compact design of the cameras themselves, and is unavoidable. It's also something you overcome fairly quickly, as you get used to the layout.

The standout feature on this case was the latch. It requires two hands to operate, making it the most difficult to open. That's a good feature. The last thing you want to do is accidentally open the case and allow the elements in. www.nikonusa.com

Sealife

Oh, my. The first unit we tested from Sealife was the Reef Master DC310 (back row, center), and it was a disappointment, to say the least. Only 3.1 megpixel, a minuscule viewfinder, no optical zoom, and a massively overbuilt case. It just wasn't elegant, took only "okay" pictures, and went through batteries at warp speed. Its one redeeming feature was the case latch. It held very well, though it wasn't enough to prevent it from owning the bottom of my list. In researching the history of digital cameras produced specifically for

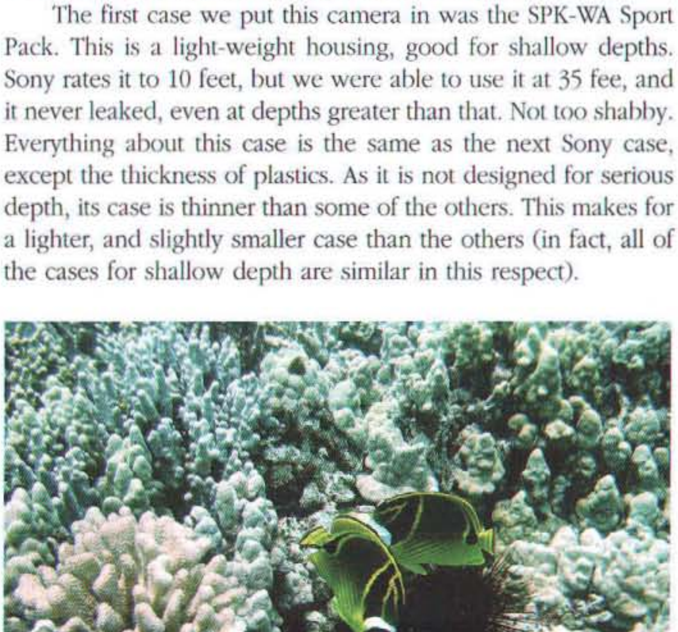
diving, we found that all manufacturers in this market tended to lag behind the mainstream by one or two generations, easily.

Then in September, Sealife released the DC500. It couldn't have been more of a 180-degree turn around from the DC310. The camera itself is a very good device. It has a five megapixel optical zoom, a large viewfinder, a rechargeable battery (with better life), and uses secure digital memory cards. The case design was head and shoulders above the previous version, with a far more compact design, a latch as good as the 310, and access to all the functions on the camera. Overall, both camera and case have good quality materials and build, particularly the case (the camera plastics feel a little flimsy), and features that almost match any of the big vendors. Sealife is still a little behind the curve, but not by much.. For some features, the Reef Master case beats out the big guys. All the buttons on the case are marked with good sized, color icons, and spaced out a bit more than other cases making it far easier to hit the right button (compared to some really dense button clusters on other cases). It's much easier to hit the right button, even with thick gloves on, and that makes a big difference. www.sealife-cameras.com

Sony

We tried out two different Sony housings, both of which held a DSC-W5 camera. This camera was a 5 megapixel unit, had a huge LCD view screen on the back (the largest of any camera we tested), used Sony's Memory Stick, was practically instant-on, and really took care of batteries. There is also a W7 7.2 megapixel version, too. On the camera side, this is a really great model to have.

The first case we put this camera in was the SPK-WA Sport Pack. This is a light-weight housing, good for shallow depths. Sony rates it to 10 feet, but we were able to use it at 35 fee, and it never leaked, even at depths greater than that. Not too shabby. Everything about this case is the same as the next Sony case, except the thickness of plastics. As it is not designed for serious depth, its case is thinner than some of the others. This makes for a lighter, and slightly smaller case than the others (in fact, all of the cases for shallow depth are similar in this respect).



Shot taken using the Sony Sport Pack.

That other Sony case we tested with the DSC-W5 was the MPK-WA Marine Pack (front row, right). It has thicker polycarbonate plastics than the SPK-WA in order to support its 40 meter depth rating. It shares many of the features of the Sport Pack, as well as most other cases, such as the button cluster on the back, the shutter and power buttons on top, as well as a wheel to control the camera mode change. The control wheel, like the one on the Nikon, works just fine. No leaks. One nice feature of the camera which makes changing modes super easy while in the Marine Pack is that it shows a circle of the mode icons on the LCD display spins them, and highlights the active one as you turn the wheel on the case. It's dead simple to change the mode on the Sony camera with that feature.

The only thing I am not fond of on these Sony cases is the latching mechanism. Though it never failed on me, I always had the concern in the back of my mind that it was going to pop open, or I was going to accidentally unlatch it while operating the camera. The mechanism latches securely, but opening and closing it is very easy to do with only two fingers, hence my concern. I don't consider it a deal breaker, because on the whole, this combination is excellent. www.sonystyle.com

Olympus

Like Sony, we had two Olympus cases to try out. The first was the PT-027 case, designed to enclose the C-7070 camera. This combination was huge. The C-7070 is an SLR sized unit, with 7.1 effective megapixels, and 4x optical zoom. It's big. Putting it in the PT-027 case made it even bigger and very unwieldy. This rig was difficult to use, especially underwater. Operation was a two-handed proposition, without exception. If you are looking to get serious about your underwater photography, you can do better than this. Look into film and the Nikonos V camera rig. It's the gold standard for pro-level photography of this type, and has been for many, many years.

The other Olympus case and camera combo we tested was the PT-026 case (front, center), which can house the Stylus 500 camera. This case is pretty decent and shares features with every other Olympus case for their other digital camera offerings. Rated to 130 ft., it was made of clear plastics, had the usual button cluster, and a decent latch mechanism. The one standout feature on the Olympus cases is the shutter button. It is a big, lever-operated contraption that is very easy to operate under any conditions. www.olympusamerica.com

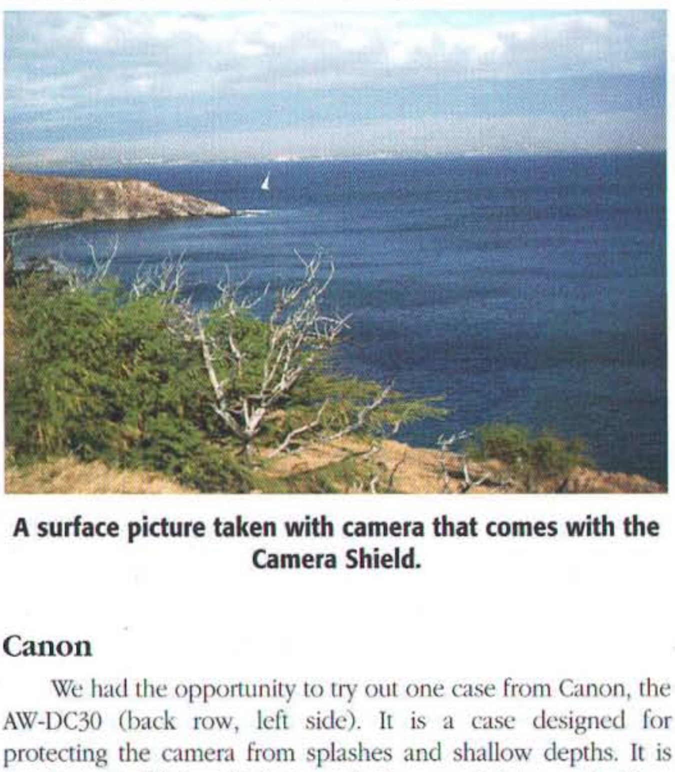
Camera Shield

Here's an interesting product. We spent some time surfing the Internet to see if we could come up with anything not widely known, or with a lot of presence, and found the Camera Shield by Fuerte Cases (front row, left in the collage above). It's essentially a box. Sporting a single opening, one side comes

completely off, through which you can place almost any camera you want. It also has one control: an adjustable lever to activate the camera shutter. The side of the box is held on with two latches, which are secured with an o-ring stretched between them. This case is utterly simple and totally brilliant.

There are two versions: one with an orange removable side for deep water, and a yellow side for a depth of 10 feet. I'm not really sure what the difference is besides color, because all the parts look the same. The yellow sided box worked just fine at depth, too (I used it on deep dives twice before I realized I had the wrong box). There is one thing to keep in mind when deciding on a Fuerte case: If you are putting your own camera in it, make sure its thickness won't be a problem. This case compresses the deeper it goes, so you need some wiggle room inside.

Don't worry if you lack a camera of your own to put in the box because the Camera Shield comes with its own. It's a basic 3.8 megapixel, camera. no optical zoom, with decent use of battery power, it accepts SD memory cards, and takes surprisingly good pictures. It's a nice addition to this very reasonable case. www.fuertecases.com



A surface picture taken with camera that comes with the Camera Shield.

Canon

We had the opportunity to try out one case from Canon, the AW-DC30 (back row, left side). It is a case designed for protecting the camera from splashes and shallow depths. It is rated to only 9.8 feet. Unfortunately the case didn't even hold up that much. A slider seal on the back of the case, through which you change the mode of the camera, failed, flooding the case in less than 5 feet of water. I was literally floating on the surface, holding the camera underneath me, when water started to leak in. Setting that aside for the moment, however, the features were fine for this type of protective case. We're just not at all sure about that seal over a sliding switch thing. It seemed an easy failure point. Your mileage may vary. www.canonusa.com

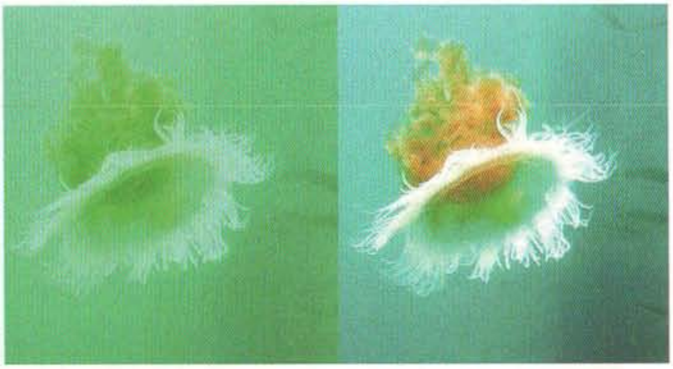
Image of a spotted eel, taken off Maui (unretouched).

In the Computer

Underwater images are notorious for their blue/green color casts, so color correction is a must (notice the green cast on the above image of an eel). No matter which image editing software you use, crop the photos to their subject matter first. I started out in iPhoto 5, where the Edit > Enhance tool produced surprisingly decent results. I clicked the Adjust button and, from their starting points, boosted the Saturation, lowered the Tint, and increased the Sharpen slider a bit, just for kicks. This gave me very reasonable results.

For serious photo editing fun, you'll need to cozy up to Photoshop or Photoshop Elements. In either program, use a

Threshold adjustment layer to sniff out which bits of the photo are supposed to black and white. Drag the Threshold slider to the left to find the blackest blacks, and right to find the whitest whites. Remember those spots and use them to set black and white points in a Curves adjustment layer. I also raised the center of the curve



A Pacific Nettle jelly, taken off Santa Cruz Island, CA: before and after (done in Photoshop).

ever so slightly in order to brighten the overall image. Last but not least, apply an Unsharp Mask. I went one step further on some of the surface images and boosted the saturation of the blue channel by adding a Hue/Saturation adjustment layer. This is a great way to bring out the brilliant blues of an ocean.

Decompress

There you have it. Our overall favorite, by far, was the Sony DSC-W5 5MP (and –W7 7.2MP) camera, and MPK-WA Marine Pack housing. This combination provided the best overall combination of features, usability, and quality of all the rigs we tested. If you are in the market for a complete set, either for yourself or as a really cool gift, Sony is the way to go.

That's not all there is to it, though. A couple of honorable mentions are needed here. The Sealife Reef Master DC 500 is a very good option if you have a SCUBA diver on your Christmas list. As an all in one solution at a reasonable price point, you really can't go wrong. Alternately, if you have someone who needs something to start with, or already has a camera and only needs something simple, the Fuerte Camera Shield is a very good option. Simple and easy to use, it's also a good gift for the occasional diver, boater, or beach goer who wants something quick and easy.

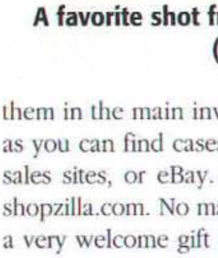
Keep in mind two things: One, except for generic cases like the Camera Shield, you need to be sure to get a case specifically designed for your camera. The tolerances inside, and the control placements really don't let you stick one camera in another case. Two, remember that just about every mainstream camera manufacturer has a line of water proof cases for almost all the cameras they sell, with varying degrees of strength.. Few companies seem to market them very well, if at all, and you'll have to dig through their web sites to find them. One other thing to note is that manufacturers only seem to keep cases for the very latest models on hand, even if they still have older cameras available. They don't seem to keep

A favorite shot from all the work done on this review (retouched in iPhoto).

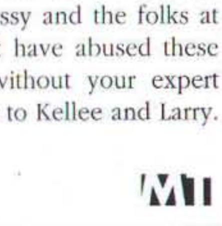
them in the main inventory for very long. Don't worry, though as you can find cases for most any camera out there on various sales sites, or eBay. We found many cases doing a search on shopzilla.com. No matter where you get the case from, it will be a very welcome gift this holiday season.

Very special thanks to Pete Sebastian and Seth, Benja Iglesis and Makena Coast Dive Charters, the Vics and Extended Horizons (all on Maui), Peace Dive Boat (in Ventura, CA), and last but not least, Blessy and the folks at B&B Scuba (also on Maui). We couldn't have abused these cases properly, nor found the shots without your expert guidance. Also, my undying appreciation to Kellee and Larry. Thanks for everything.

About The Authors



Michael R. Harvey is the reviews editor here at MacTech. He's only almost died twice while diving, and while he is bent, that's more a mental health issue than poor dive planning. Questions, comments, and directions to good Southern California dive sites can be sent to him at reviews@mactech.com.



Lesa Snider writes weekly tutorials for the National Association of Photoshop Professionals, PlanetPhotoshop.com, and Layers Magazine. She is also New York Times Technology columnist David Pogue's personal assistant, and screen shot goddess/technical editor of several Missing Manual books. Lesa's web site, www.graphicreporter.com, provides tips, tutorials, and reviews for all levels of creative enthusiasts.

Aaron Adams is a network administrator, consultant, former star of Apple's "Switch" campaign, and an avid Guinness drinker. He can be reached via e-mail at adamsa@mac.com.

Building a Table View Project in AppleScript Studio

Back in my April column, I provided an introduction to AppleScript Studio, a technology that is a part of Xcode and Interface Builder, Mac OS X's powerful integrated development environment. Using AppleScript Studio, AppleScript developers are able to create fully native Mac OS X applications, complete with user interfaces. By constructing scripts with interfaces, developers can build complex and powerful user friendly AppleScript solutions that have the same look and feel of any other Mac OS X application.

In addition to building stand-alone AppleScript solutions, AppleScript Studio can also now be used to create Automator actions, for use with Mac OS X 10.4's popular new Automator application. In my August column, I walked through the process of building of a simple AppleScript Studio-based Automator action.

This month, we will create a new AppleScript Studio-based project that is a little more complex, and incorporates a table view into its interface. This application, which I call ***Enable|Disable Mail Accounts***, will display a list of email accounts in Mail to the user, and allow the user to toggle whether the accounts are enabled or disabled.

Table Overview

Before we get started, I would like to briefly discuss the concept of tables. In AppleScript Studio, a table, which is actually referred to as the class ***table view***, may be used to display a list of records, complete with field values, similar to what you might see in a spreadsheet. See figure 1.

Figure 1. An Example of a Table View

While a table view can be placed onto an interface window, or any type of view class for that matter, within AppleScript Studio's class hierarchy, a table view is actually not a direct element of the window itself. Rather, it is an element of a scroll view, which typically is an element of the window. Because of this, when referring to a table, you must do so within its parent scroll view. For example:

```
table 1 of scroll view 1 of window 1
```

Much like a spreadsheet, a table view may be configured to display multiple rows of data, in one or more columns. When

populating or manipulating the contents of a table view, the preferred and suggested method of doing so is through the use of what is known as a data source. A data source is essentially equivalent to a virtual database within your AppleScript Studio project, complete with records and fields that can hold the values for your table. A data source, which can be created up front in your project, or dynamically through AppleScript code, may be linked to a table view. Once linked, the table view's visual contents on the window will automatically reflect the contents of the data source. So, in other words, as you make changes to the contents of the data source, those changes will automatically translate directly to the linked table view.

There are other ways to populate and manipulate data within a table view, without creating or interacting with a data source. I won't be covering those methods in this month's column, but I want you to be aware that there are multiple ways of accomplishing the same goal when working with table views. However, as I said before, using data sources is the preferred method of populating a table view, as it is typically the most efficient and has some benefits that other methods do not. For example, when using a data source, it is possible to suspend updating of the table view during modification of the data source. This updating can then be re-enabled once the data source is finished being modified, thus allowing the table view's display to update all at once.

Working with table views, scroll views, and data sources can seem complex at first. As we proceed through the remainder of this month's column, hopefully, these classes will begin to make more sense to you. However, for more detailed overviews of these classes, you may want to consult the ***AppleScript Studio Terminology Reference***, which is included with Xcode's documentation, and can also be found in the ADC Reference Library on the Apple Developer Connection website at http://developer.apple.com/.

Now, let's get started on our project.

Building the Project

The first step in building our project is to create a new AppleScript Studio project. As always, begin by launching Xcode. Please note that, for this column, my project was developed using Mac OS X 10.4.2 and Xcode 2.1. That said, the code included in this column may or may not function in older versions of the OS and Xcode. However, please be aware that new software versions often result in changes in AppleScript terminology. Therefore, if you are using software versions other than those that I have specified, your required terminology may differ slightly from that which I am using in this article.

In Xcode, create a new project, by selecting ***New Project...*** from the ***File*** menu. When prompted, select a project type of ***AppleScript Application*** from the list of available project templates, specify a project name of ***Enable|Disable Mail Accounts***, and specify an output folder for the project. Xcode will duplicate the ***AppleScript Application*** project template into the specified output folder, and open it before you. See figure 2.

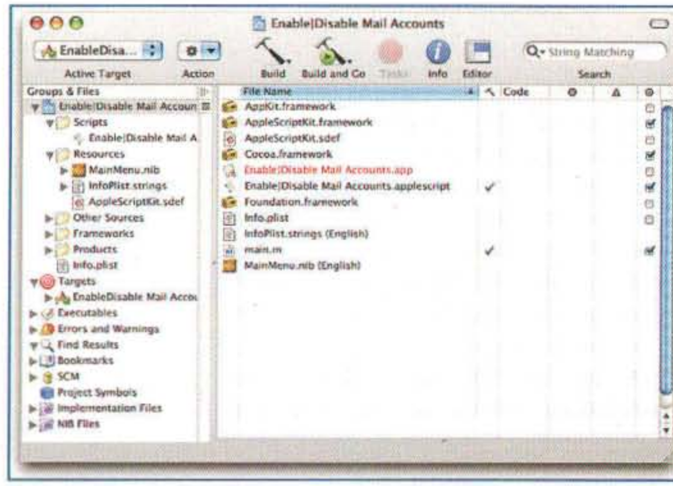


Figure 2. Enable|Disable Mail Accounts Project Window

Building the Interface

The next step in the creation of our AppleScript Studio project will be to design the interface. To do this, double click on the ***MainMenu.nib*** file within your Xcode project window. This will open the project's default view in the Interface Builder application.

Designing the Window

By default, the project's interface should already contain an empty window view. This window will be the basis for our interface. Click on the window, and give the window a name by entering ***Enable|Disable Mail Accounts*** into the ***Window Title*** field in the ***Inspector*** palette. See figure 3. If the ***Inspector*** palette is not visible, simply select ***Show Inspector*** from the ***Tools*** menu in Interface Builder.

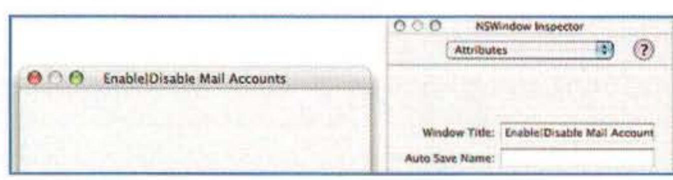


Figure 3. Assigning a Window Title

While the window is still selected, also de-select the ***Visible at Launch Time*** checkbox, if it is selected. This will cause the window to be hidden when the project is first launched. This means that, later, we will need to add code to display the hidden window. The reason we want it to be hidden initially is so that we can populate a table view with data, prior to displaying the window.

Next, click on the *Cocoa Data Views* tab in the toolbar of the *Palettes* window. If the *Palettes* window is not visible, select *Palettes > Show Palettes* from the *Tools* menu. Next, locate an *NSTableView* interface element in the *Palettes* window, and drag it into your project's window. See figure 4.

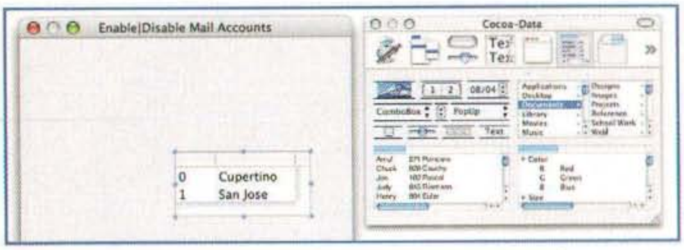


Figure 4. Adding an NSTableView

Using the *Inspector* palette, configure the scroll view to contain two columns. Next, double click on the scroll view to select the table view element. Finish preparing the table view by clicking on each column within the table view, and entering header titles of *Account Name* and *Status* for the column headers. See figure 5 for an example of how the table view should appear, once it has been configured with two columns, complete with column headers.

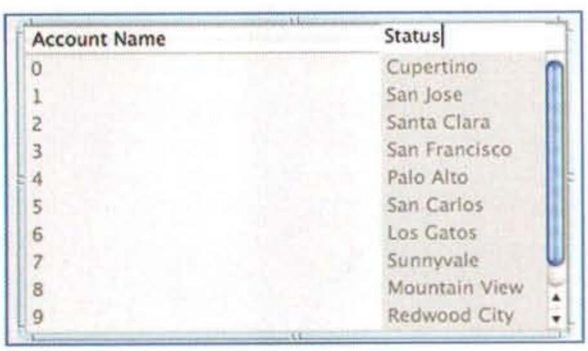


Figure 5. Assigning Table View Column Header Titles

Next, click on the *Cocoa Controls and Indicators* button in the toolbar of the *Palette* window, and drag two buttons into your interface window. Enter *Quit* and *Toggle* as the titles for these buttons in the *Inspector* palette.

Arrange the interface elements in the window, and make any necessary adjustments until the window resembles the example shown in figure 6.

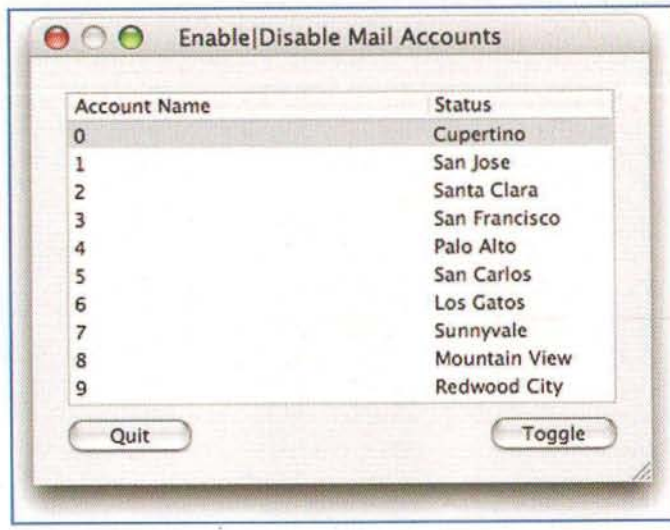


Figure 6. The Complete Solution Interface

As you design an interface, always try to adhere to Apple's standards for human user interface guidelines, which can be found in the ADC Reference Library, both online and in Xcode's documentation.

Preparing the Interface for AppleScript Interaction

Once you have finished designing the interface, the elements that make up the interface must be prepared for the AppleScript code within your project to interact with them. This task will consist of assigning AppleScript names to various interface elements, as well as configuring certain elements of the interface to respond to event handlers.

First, we will assign an AppleScript name to the main window itself. To do this, click on the window to select it. Next, choose *AppleScript* from the popup button at the top of the *Inspector* palette, and enter the name *Main Window* into the *Name* field. See figure 7. The process of assigning AppleScript names to other interface elements will be the same.

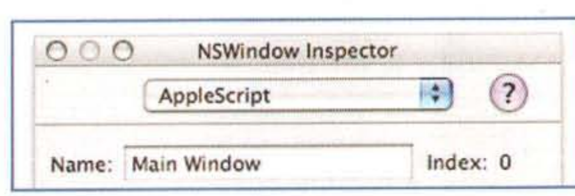
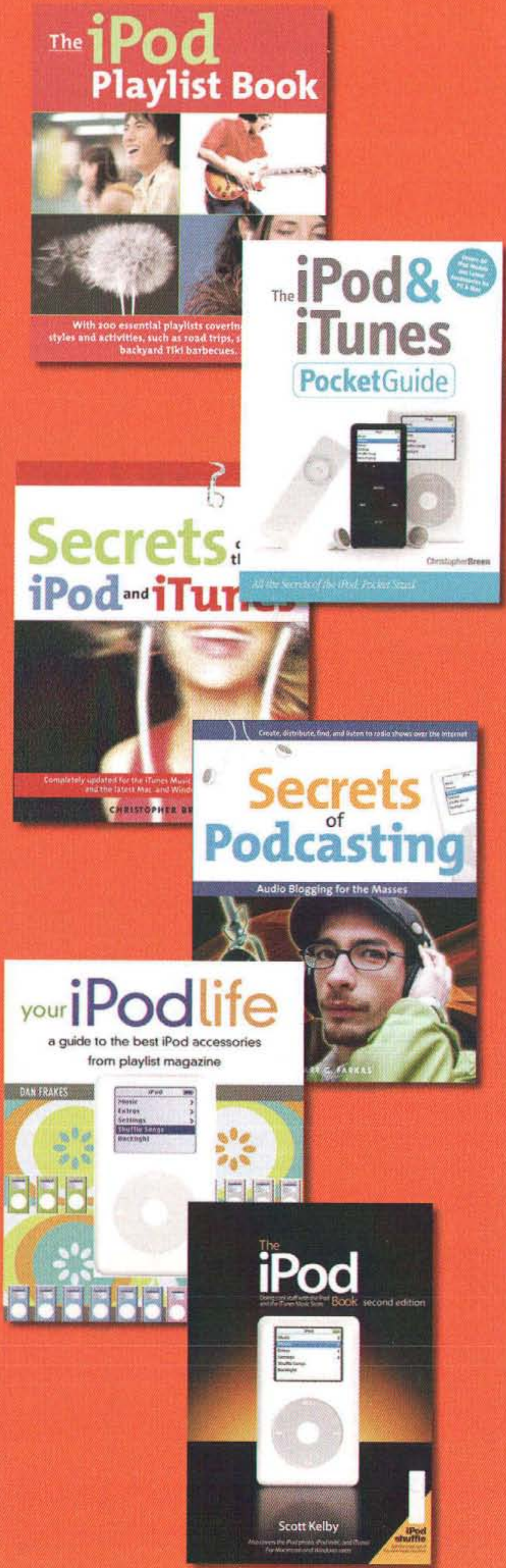


Figure 7. Assigning an AppleScript Name to the Main Window

Next, select the *Quit* and *Toggle* buttons that were placed on your window, and assign AppleScript names of *Quit* and *Toggle* to these buttons, respectively. For each of these buttons, we will also assign an event handler. A listing of the event handlers to which an interface element can respond are located on the *Inspector* palette, beneath the *Name* field. For these two buttons, enable the *clicked* event


The iPod Playlist Book
The iPod & iTunes PocketGuide
Christopher Breen
Secrets of the iPod and iTunes
Secrets of Podcasting
Audio Blogging for the Masses
your iPod life
a guide to the best iPod accessories
from playlist magazine
DAN FRAKES
The iPod Book second edition
Scott Kelby

handler by selecting the appropriate checkbox. You will then need to link the event handler to the AppleScript file within your project code. So, select the checkbox next to ***Enable|Disable Mail Accounts.applescript***, in the ***Script*** area at the bottom of the ***Inspector*** palette, beneath the event handler list. See figure 8 for an example of a properly configured button.

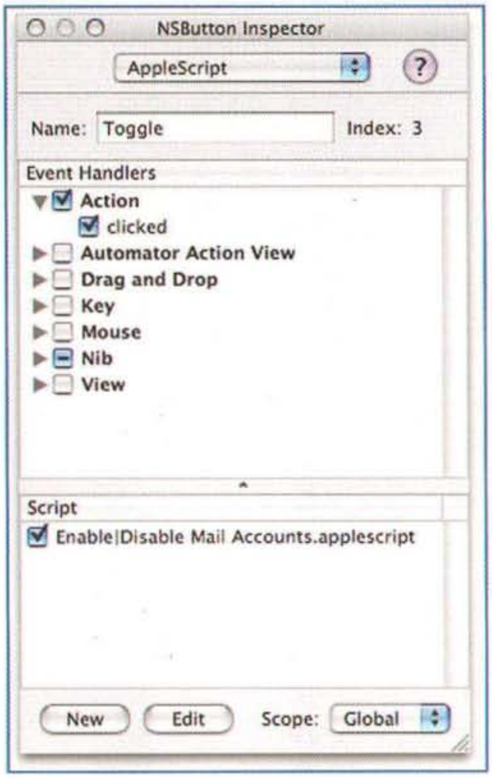


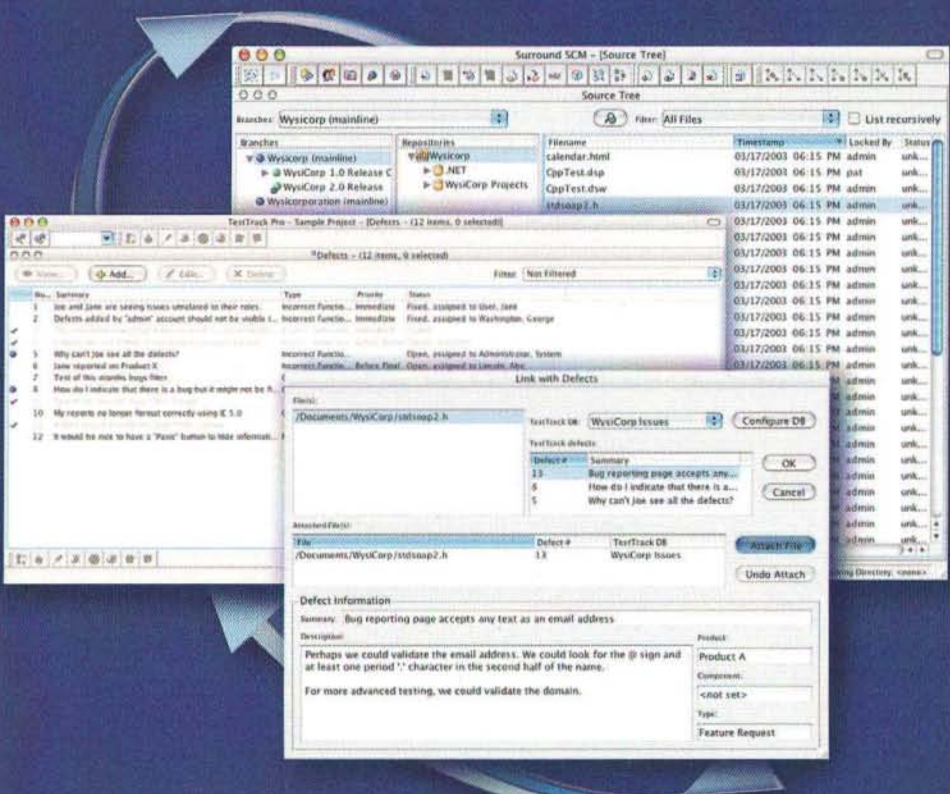
Figure 8. Configuring a Button

Select the scroll view in the window, and assign an AppleScript name of ***Account List*** to the scroll view. Next, double click on the scroll view to select the enclosed table view. In the ***Inspector*** palette again, assign an AppleScript name of ***Account List*** to the table view. Also, enable the ***Awake from Nib*** event handler for the table view, and link it to the AppleScript in your project. See figure 9.

Figure 9. Configuring the Table View

Complete Source Control and Defect Management for Mac OS X

Seapine Software™
changing the world of software development



Effective source code control and defect tracking require powerful, flexible, and easy-to-use tools—Surround SCM and TestTrack Pro

- Complete source code control with private workspaces, automatic merging, role-based security, and more
- Comprehensive defect management — track bug reports and change requests, define workflow, customize fields
- **New!** Full Unicode support maintains international characters across various languages, ensuring data is not lost or misinterpreted
- Advanced branching simplifies managing multiple versions of your products
- Fast and secure remote access to your source files and defects — work from anywhere
- Scalable and reliable cross-platform, client/server solutions support Mac OS X, Windows, Linux, and Solaris
- Exchange data using XML and ODBC, extend and automate with SOAP support
- Licenses priced to fit your budget

Seapine Software Product Lifecycle Management
Award winning, easy-to-use software development tools

Seapine **Surround SCM**

Seapine **TestTrack** *PRO*

software development 13th annual product excellence award JOLT

*Download Surround SCM and TestTrack Pro at **www.seapine.com/mac** or call 1-888-683-6456*

With the table view still selected, click on each of the ***Account Name*** and ***Status*** column headers, and assign AppleScript names of ***Account Names*** and ***Status*** to each of these headers, respectively, using the ***Inspector*** palette. No event handler assignments are necessary for these elements.

Finally, as a last step in configuring the interface, select ***File's Owner*** in the ***MainMenu.nib*** window, enable the ***launched*** event handler, and link it to the AppleScript in your project, using the ***Inspector*** palette. See figure 10.

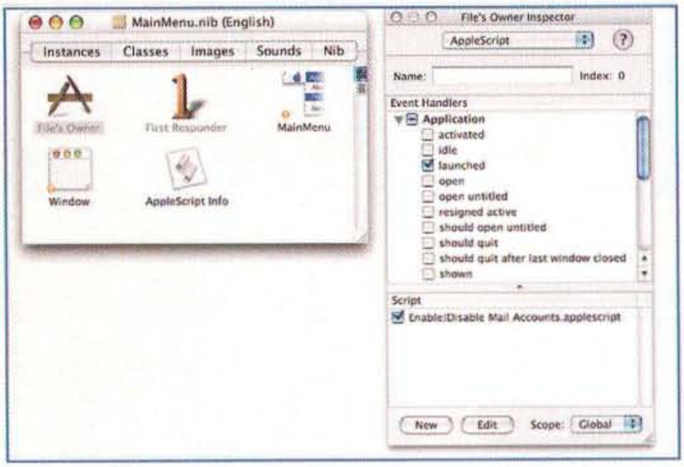


Figure 10. Enabling the Launched Event Handler for the Solution

Adding the AppleScript code

Now that you have configured your interface, return to Xcode again, as it is time to begin adding the AppleScript code into your project. Double click on the ***Enable|Disable Mail Accounts.applescript*** file in your project to begin editing the AppleScript code.

Building the Data Source

The first AppleScript code that we will add to the project will be code that creates a data source, which will later be populated to update our table view's contents. We want this code to be triggered automatically, prior to actually displaying the interface. Therefore, to do this, we will make use of the `awake from nib` handler, which will be triggered by the project when the interface is first loaded.

Type the following code into your project:

```
on awake from nib theObject
  set theDataSource to make new data source at end of data
sources with properties {name:"Account Data"}
  tell theDataSource
    make new data column at end of data columns with
properties {name:"Account Name"}
    make new data column at end of data columns with
properties {name:"Account Status"}
  end tell
  set data source of theObject to theDataSource
end awake from nib
```

The first line of code within this handler will create a new data source, named ***Account Data***. The next few lines will create columns for the account name and status within this data source, to match our table view configuration.

In our interface, we configured our table view with an `awake from nib` event handler. Therefore, when this handler is triggered, a reference to the table view will be passed to the handler in the parameter named `theObject`. The final line of code within the handler will set the `data source` property of the table view to the newly created data source, thus linking the table to the data source. Once linked, any changes we make to the contents of the data source will be reflected in the table view.

Refreshing the Table

The next code we will add to our project is a custom handler, which I have named `refreshAccountListing`. This handler will be called by other code within our project, as you will see shortly. The responsibility of this code will be to populate the contents of the data source that will be built with the previous code we discussed.

Type the following code into your project:

```
on refreshAccountListing()
  tell application "Mail"
    set theAccountNameList to name of every account
    set theAccountStatusList to enabled of every account
  end tell
  tell data source "Account Data"
    set update views to false
    delete every data row
    repeat with a from 1 to length of theAccountNameList
      set theRow to make new data row at end of data rows
      tell theRow
        set contents of data cell "Account Name" to item a
of theAccountNameList
        if item a of theAccountStatusList = true then
          set theAccountStatus to "Enabled"
        else
          set theAccountStatus to "Disabled"
        end if
        set contents of data cell "Account Status" to
theAccountStatus
      end tell
    end repeat
    set update views to true
  end tell
end refreshAccountListing
```

This handler will first retrieve the names and statuses of any email accounts in Mail. Next, the handler will tell the data source to set its `update views` property to a value of `false`. This will actually cause the table view to cease updating temporarily, while the data source is being modified.

Next, the handler proceeds to delete any existing data rows from the data source. Within a data source, a `data row` represents a single row of data within the table. A data row contains `data cells`, which are essentially fields to hold the data, which translate to the columns within a table view. The reason we want to delete the data rows within the data source first is because we will trigger this handler any time we want to refresh the contents of the data source. Before doing this, we need to delete any existing rows that may have been previously created, so that the data source is completely refreshed, and new data is simply not appended to existing data.

After deleting any existing data rows, the handler will then loop through the retrieved email account names, and create a new data row for each, inserting the account name and its corresponding status into the appropriate data cells for each row.

Finally, the `update views` property of the data source is set to a value of `true`, allowing the table view to update to reflect the new contents of the data source.

Initializing the Table

Next, we are going to add the code that will initially trigger the `refreshAccountListing` handler to populate the data source. Once populated, this code will also make the main window visible to the user.

Type the following code into your project:

```
on launched theObject
  refreshAccountListing()
  set visible of window "Main Window" to true
end launched
```

This `launched` handler will be triggered automatically when the project is run, immediately following the successful execution of the `awake from nib` handler.

Toggling the Email Account Status

Next, we need to add code to actually toggle the status of any selected email accounts in the table view. To do this, type the following code into your project:

```
on toggleAccounts()
  set theSelectedRows to selected data rows of table view
"Account List" of scroll view "Account List" of window "Main
Window"
  repeat with a from 1 to length of theSelectedRows
    set theRow to item a of theSelectedRows
    tell theRow
      set theAccountName to contents of data cell "Account
Name"
      set theAccountStatus to contents of data cell "Account
Status"
      if theAccountStatus = "Enabled" then
        set theNewAccountStatus to false
      else
        set theNewAccountStatus to true
      end if
      tell application "Mail"
        set enabled of account theAccountName to
theNewAccountStatus
      end tell
    end tell
  end repeat
  refreshAccountListing()
end toggleAccounts
```

This handler will be triggered when the user clicks the *Toggle* button in the interface. It will retrieve a listing of any selected accounts in the table view, toggle the status of those accounts in Mail, and then trigger the `refreshAccountListing` handler to update the table view to reflect the newly modified account statuses.

Adding Button Code

Finally, we need to add code to perform the appropriate task when a button in the project's interface is clicked. To do this, type the following code into your project:

```
on clicked theObject
  if name of theObject = "Toggle" then
    toggleAccounts()
  else
    quit
  end if
end clicked
```

If you recall, we applied a `clicked` event handler to each of the buttons in our interface. Therefore, this handler will be triggered whenever the user clicks one of those buttons. This handler will check the name of the button that was clicked. If it was the *Toggle* button, then the handler will trigger the `toggleAccounts` handler. If not, it assumes that the user clicked the ***Quit*** button, and it simply quits the application entirely.

Testing the Project

Now that our project is complete, it is ready for testing. To test the project, select ***Build and Run*** from the ***Build*** menu in Xcode. If everything works as expected, your solution should launch, and the interface should be displayed, complete with a table view showing your Mail account names, along with their statuses. See figure 11.

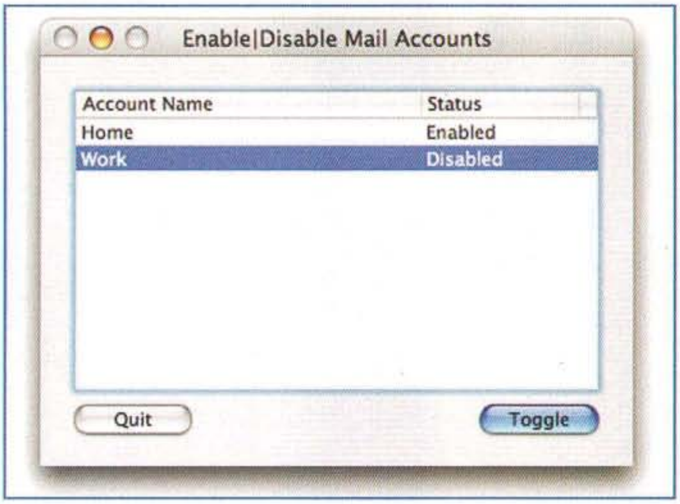


Figure 11. The Completed Application

Select one or more accounts in the table view, and click the *Toggle* button to verify that the account statuses are toggled in Mail, and the project properly functions.

In Closing

AppleScript Studio provides many possibilities for creating interfaces. As you can see from this relatively simple interface, with the use of table views, you can construct interfaces for your AppleScript Studio projects that will display information to the user in rows and columns, and even allow a user to make selections from this data.

For example, a table view might list file names, email addresses, database fields, etc. The possibilities are virtually limitless. What's more, you can even extend the functionality of table views further by adding embedded buttons, checkboxes, and more to the columns within the table view. See figure 12 for an example of this. Using this type of technique, we could update the *Enable|Disable Mail Accounts* project to allow a user to toggle email account statuses by making selections from individual popup menus, rather than clicking a *Toggle* button.

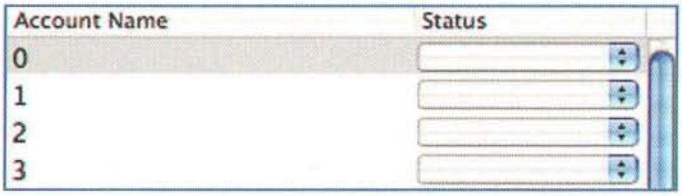


Figure 12. An Example of Embedded Interface Elements within a Table View

Please note that I have made the sample project discussed in this article available for download from my web site at the following URL:

http://www.automatedworkflows.com/files/demos/MacTECH.12.05.Example.zip

I encourage you to continue exploring table views on your own, and expanding the sample project discussed in this month's column to include even greater functionality.

Until next time, keep scripting!

MT

About The Author

Ben Waldie is author of the best selling books "AppleScripting the Finder" and the "Mac OS X Technology Guide to Automator", available from http://www.spiderworks.com. Ben is also president of Automated Workflows, LLC, a company specializing in AppleScript and workflow automation consulting. For years, Ben has developed professional AppleScript-based solutions for businesses including Adobe, Apple, NASA, PC World, and TV Guide. For more information about Ben, please visit http://www.automatedworkflows.com, or email Ben at applescriptguru@mac.com.

What's Rattling Around in My Sed

Automating Edits.

"OS X is Unix" – a statement that has been under a bit of debate since the OS shipped. It does look like a duck and does smell like a duck, but it doesn't ***really*** quack like a duck. While many, many projects will, unmodified, compile and run under OS X, there are enough that do not thanks to the acrobatics that OS X engages in underneath it all. That's why we all have to delight in any project that comes pre-installed for us – no compile necessary, and we know it'll be on all OS X boxes that we touch – ***and*** runs identically to the Solaris/IRIX/Linux/HP versions. There are two indispensable utilities that fall into this category: sed and awk. These will consume us the next few columns. With sed ("the stream editor") being the more simple of the two, that's where we'll start.

Cuckoo for Cocoa Puffs!

OK – it's not ***that*** kind of cereal! sed is a "serial editor" as in it will repeatedly make changes to an input stream in series. I was, not all that long ago, amongst a group of like-minded Mac techs and one person asked how he could take a single file and alter it many times over for specific destinations. I immediately offered sed as a solution. "What?" "sed," I repeated. Blank stares. sed and awk are too useful to not use!

sed takes input, via file name or standard input, processes the file and pumps the results back via standard out. However, instead of making edits interactively, like you might with vi or a word processor, sed allows you to script your edits, almost as if you're creating a macro. This script can then be applied to many (or, any) files. Just like your editor has commands that will insert a line, delete a line and so on, so does sed. Even more impressive, is sed's ability to alter a file, or stream through substitution and regular expressions. And there's the rub. We can 't talk about sed without talking about regular expressions (RE).

Lucky Charms

Regular expressions are fundamental to the Unix landscape. Understanding them makes your computing life better and easier on many levels. Even many GUI editors have support for RE built in. Notable examples are BBEdit and Dreamweaver. There are patterns that you can describe with RE that just can't be done cleanly otherwise. A full discourse on regular expressions is its own article or even book. Fortunately, there's plenty of material out

there. The February 2005 issue of MacTech ran, "Matchmaking With Regular Expressions" by Paul Ammann, so if you've been a subscriber, or have that issue, you should start there. O'Reilly has an entire ***book*** on RE alone called "Mastering Regular Expressions." If you've already mastered RE, please skip ahead to the next section. Otherwise, I'm ready to present the world's shortest intro and tutorial on regular expressions.

Regular expressions are one of those things that can be a little daunting at first, because the syntax is a little out of the ordinary. However once you dive in, you'll start seeing patterns everywhere that are a great fit. I will admit that I'm going to gloss over some of the finer details regarding RE.

A regular expression is just that: an expression. It's not to be taken literally as it is by itself. This is because most patterns involve metacharacters. We all intrinsically now understand the wildcard metacharacter, "*". If we were to type into our shell, "ls –l *", we'd expect a listing of all files, as we're not literally looking for a file named "*". Similarly, sed is heavily dependant on metacharacters in RE. Also, as an expression, it is something that the interpreter evaluates, much like a mathematical expression. Since understanding how the interpreter evaluates each metacharacter is fundamental to your understanding of the concepts covered in this piece, that's where we'll start.

Non-metacharacters are matched verbatim. If you ask sed (or grep) to match "Cat", it will match a capital "C" immediately followed by a lower case "a", adjacent to a "t". No other permutation will match. Each character is its own RE that matches that single character. To match any single character, use the "." (dot) meta. So, a regexp of "ca." would match "cat" or "cab", but not "rat". To match a wider swath of "any" characters, use the wildcard "*" (asterisk). The asterisk behaves a little differently than you may think. It ***modifies*** the previous character, and matches ***zero or more*** characters. Again: it ***modifies*** the previous character. If you are looking for "car", a RE of "c*r" will not do what you expect. Yes, it will match "car", but will ***also*** match ***any*** word with "r" in it. Huh? "c*r" says, "match zero or more of the letter c, and then an r." That re-enforces the second point: it will match ***zero or more*** characters. To find words that begin with "c" and end with "r", you can use "c.*r", which will match "cr" (if that was a word), "car" and "cur", but also "choir", "czar", "carpetmonger", "calcaneoplantar"...and much more, as we'll see. This says, "match the letter 'c', zero or more of any character, and then an 'r'".

Well, if you actually ran the above on a dictionary file, you'd notice that you'd get a ***lot*** more that expected. How does "c.*r" match "buckaroo"? We need to learn two more metacharacters: beginning-of-line "^" (circumflex) and end-

of-line "$" (dollar sign). So, to truly get words that begin with "c" and end with "r", you'd use a RE of "^c.*r$". That says, "at the beginning of the line, look for a 'c', then match zero or more characters, and finally match an 'r' at the end of the line."

Cheerios

Let's jump right in with an example, shall we? Probably the sed command (or, ***mnemonic***) used most often is substitute – "s":

```
sed -e 's/Mike/Michael/' letter.txt
```

This tells sed to run through the file letter.txt and replace 'Mike' with 'Michael'. Note the use of the forward slash as a ***delimiter***. However, if you read the preceding section, you may have come to expect that it's never quite that simple. The substitute command will only make the substitution for the first occurrence on a line, unless you add the global flag – 'g'. Simply, the command should be:

```
sed -e 's/Mike/Michael/g' letter.txt
```

That'll get all of them. You can even make multiple changes in a file:

```
sed -e 's/Dot/Dorothy/g' -e 's/Chris/Christopher/g'
personnel.txt
```

This gives us an opportunity to talk about how sed applies edits, which is critical to understanding what's going on.

sed works with a line that it brings into its ***pattern space***. It then applies ***all*** edits that you've asked for to this line, moves the updated line to stdout (if appropriate), and then brings the next line into the pattern space. To really understand this, a demonstration is in order. The following text is in a file called 'short_story.txt':

> Bill and Michael went to the store. Bill needed to buy some butter, eggs and flour. He and Michael were in a hurry to bake a cake for their parent's Anniversary. Once they got home, Bill and Michael realized that they forgot cake icing.

Of course, Michael, being the older brother, feels he should precede Bill in the story. Well, you have sed and that's an easy task, right? Wherever you see 'Michael', change it to 'Bill', and wherever you see 'Bill', change it to 'Michael':

```
$ sed -e 's/Bill/Michael/g' -e 's/Michael/Bill/g'
short_story.txt
```

But when you run the command, you get this output:

> Bill and Bill went to the store. Bill needed to buy some butter, eggs and flour. He and Bill were in a hurry to bake a cake for their parent's Anniversary. Once they got home, Bill and Bill realized that they forgot cake icing.

What happened? Let's trace:

1. sed brought the first line into pattern space.

2. sed found the pattern 'Bill' and substituted 'Michael', making the line:
"Michael and Michael went to the store. Michael needed to buy"

3. sed then applied the second edit we asked for, making 'Michael' into 'Bill', resulting in the output we just saw.

Be aware of that interaction. Later on (read: next month's column), I'll show how to deal with that. Using the simple substitution above is great static files, like a checklist:

```
Hello -=firstname=-! Welcome to Acme and Associates.
There are some things you'll need to know to get
started with our network. Please keep a record of
these items:
Name: -=username=-
Phone: -=phonenum=-
IP address: -=ipaddr=-
Thanks!
```

You could write a sed script that alters this file appropriately before mailing it out. Of course, substitution gets much more interesting when combined with regular expressions. But first, we need to look at some of the other sed commands.

Rice Krispies

As mentioned, sed is an editor. What good would an editor be if it couldn't add and delete lines? The simpler of the two is delete. Delete erases the entire pattern space, and then continues on to the next line – there's nothing left to match after that.

```
sed -e '1d' short_story.txt
```

This prints out our short story, minus the first line – that gets deleted. Of course, delete is even more powerful when combined with regular expressions. How about one that gets rid of bash comments:

```
sed -e '/^#/d' bashscript.sh
```

To create more complex sed interactions, we can put all of our commands in a file to make a sed script. By placing your edits into a sed script, you gain the advantage of making your routine reusable, and being able to build it up slowly – these scripts can get complex very quickly, and it's not often that you get it 100% right on the first shot. If you were to save the following into a file named "editscript.sed":

```
1d
s/Bill/William/g
s/cake/pie/g
```

...you'd be able to run it against our short story, using the "-f" flag, and see each edit:

```
$ sed -f editscript.sed short_story.txt
some butter, eggs and flour.  He and Michael were in
a hurry
to bake a pie for their parent's Anniversary.  Once
they got
home, William and Michael realized that they forgot
pie icing.
```

First line deleted, "Bill" becomes "William" and "cake" becomes "pie". Using a sed script also brings some other advantages in terms of opening up other commands for our use. First and foremost, as opposed to delete, we can insert. Insert is one of the odd sed commands in that it expects its input to be broken up over more than one line. If we wanted to insert a title at the top of our story, we could use this:

```
1i\
The Wedding Cake
```

With insert, you ***must*** include a backslash after the command with the text to be inserted on the next line.

Also, within a sed script, we can use functions. ***Functions*** in sed? Well, perhaps they're not like traditional functions, but we can opt for a series of edits to take place on a sub-set of the document we're editing:

```
/^    Bill/ {
      s/Bill/William/
      s/    /	/
}
```

(note here that the second edit is "s slash space space space space slash tab slash". Thank you for your patience.)

This tells sed, "only on lines that start with four spaces immediately followed by 'Bill' will you make the following edits: change Bill to William and then change four spaces to a tab character." Think of all the conditional ways you can program edits with this feature! (OK, did I get too excited over that?)

Again, the power of regular expressions increases sed's value exponentially. This is especially effective when dealing with any kind of mark-up. And to make this even more powerful, we need to introduce addressing. So far, our commands have not had an address, which makes sed apply edits to each line. Another possibility is to supply one address. This can come in the form of a number, or a pattern. If we only wanted to make substitutions on lines that end with a period, we could use something like this:

```
sed -e '/\.$/s/\./!/g' short_story.txt
```

(The address to affect is in bold)

This tells sed, "only on lines that end with a period ("\." – we have to escape the period, otherwise it'll match any

single character), substitute an exclamation point for any period on the line." OK, it's a completely contrived example, but it should convey the power this brings. I used a not so contrived example the day I finished this article. I was working on a mailing for a client (not a spammer), that needed to retrieve their client e-mail addresses from a Crystal Reports CSV file. CR writes crummy CSV, only enclosing fields in quotes if they have an embedded comma. While that's valid, the software this client was using for mailing didn't like that format too much. sed to the rescue! Here's what I did:

```
$ sed '/^\"/s/,/ /1' clients.csv > clients2.csv
```

This told sed, "only on lines that begin with a quote (gotta escape the quote there), substitute a space for the first comma that you find." This output was then redirected to another file, which we had the mailing software (happily) read in.

Don't feel confuSED

Just learning the sed presented in this month's column will bring you incredible power on the command line and when editing files in batch or under the command of a shell script. Believe it or not, there's more! If this was your first exposure to sed, practice, practice, practice! Get a test file (or two or three) and see what sed does when you issue various commands. Next month, I'll cover a little more on the quirky interactions, other commands and more advanced usage.

While I normally say, 'see you next month,' and I mean 'in print,' I really ***do*** hope to see everyone next month in person: at MacWorld! I'll be there all week, and presenting the, "From the Chime to the Desktop" session on Wednesday in the IT track conference. Please say hello if you'll be in San Francisco! Of course, I'll still see everyone in print. Until then, let sed rip through some files for you!

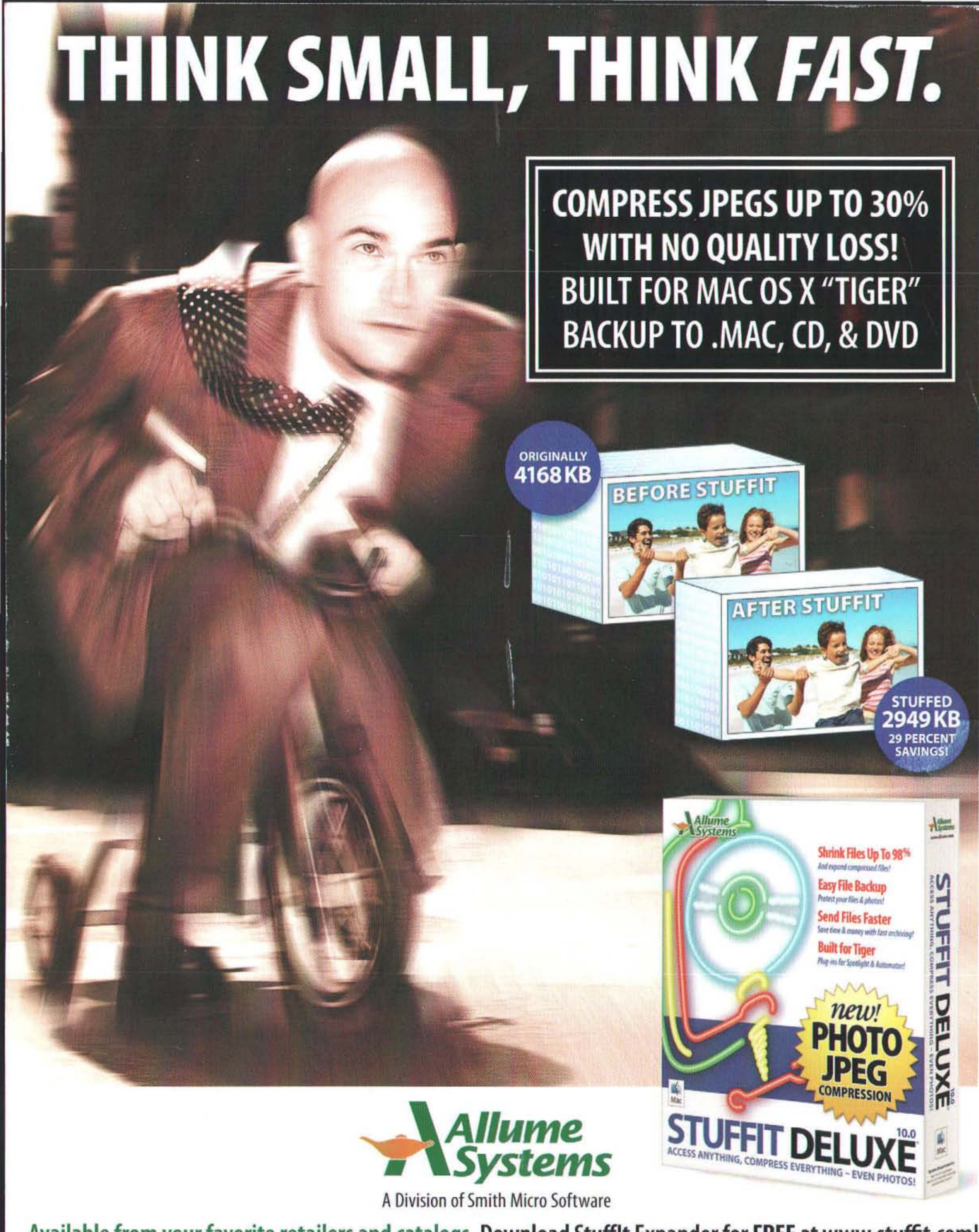
MT

About The Author

Ed Marczak owns and operates Radiotope, a technology consulting company. Ed will be speaking and hanging out at MacWorld SF 2006 all week – hope to see you there! He also deposits tech thoughts on-line at http://www.radiotope.com

Advertiser/Product Index

The index on this page is provided as a service to our readers. The publisher does not assume any liability for errors or omissions.


THINK SMALL, THINK FAST.
COMPRESS JPEGS UP TO 30%
WITH NO QUALITY LOSS!
BUILT FOR MAC OS X "TIGER"
BACKUP TO .MAC, CD, & DVD
ORIGINALLY
4168 KB
BEFORE STUFFIT
AFTER STUFFIT
STUFFED
2949 KB
29 PERCENT
SAVINGS!
Allume Systems
Shrink Files Up To 98%
And expand compressed files!
Easy File Backup
Protect your files & photos!
Send Files Faster
Save time & money with fast archiving!
Built for Tiger
Plug-ins for Spotlight & Automator!
new!
PHOTO
JPEG
COMPRESSION
STUFFIT DELUXE 10.0
ACCESS ANYTHING, COMPRESS EVERYTHING – EVEN PHOTOS!
Allume Systems